AD692836

APPLICATIONS OF DISCRETE OPTIMIZATION TECHNIQUES TO CAPITAL INVESTMENT AND NETWORK SYNTHESIS PROBLEMS

by

Felipe Ochoa-Rosso

SEARCH AND CHOICE IN TRANSPORT SYSTEMS PLANNING

Volume III of a Series

Prepared in Cooperation with

the M.I.T. Urban Systems Laboratory

R68-42

June 1968

MIT

DEPARTMENT OF CIVIL ENGINEERING

SCHOOL OF ENGINEERING
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Cambridge, Massachusetts 02139

Research Report R68-42

APPLICATIONS OF DISCRETE OPTIMIZATION TECHNIQUES TO CAPITAL INVESTMENT AND NETWORK SYNTHESIS PROBLEMS

by

Felipe Ochoa-Rosso

Assistant Professor of Civil Engineering (Visiting)

SEARCH AND CHOICE IN TRANSPORT SYSTEMS PLANNING

Volume III of a Series

Prepared in cooperation with
the M.I.T. Urban Systems Laboratory

Sponsored by the U.S. Department of
Transportation and the General Motors Grant
for Highway Transportation Research

Transportation Systems Division
Department of Civil Engineering
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

June, 1968

ABSTRACT

The purpose of this work is to formulate and solve certain optimization problems arising in the fields of engineering economics, scarce resource allocation, and transportation systems planning.

The scope and structure of optimization theory is presented in order to place subsequent work in proper perspective. A branch and bound algorithm is rigorously developed which can be applied to the optimization problems of interest. A rounding operation is defined, which provides a powerful rejection rule and permits the calculation, at each stage of the solution process, of an upper bound and a feasible solution in addition to the usual lower bound. This double bounding technique implies little or no extra computational effort.

Subsequent chapters are devoted to the study of various cases of capital investment problems. Investment in sets of independent projects is considered first. For the (0-1) multi-dimensional knapsack problem a new formulation, interpreted as a network synthesis problem on a bipartite graph, is given. This formulation permits the straightforward application of the branch and bound algorithm, and allows the solution of the linear program associated with each node of the solution tree to be obtained by inspection.

This study is pursued by considering capital investment in a single time period as a special case of the previous problem. Certain economic interpretations are derived by investigating the dual program of the discrete knapsack problem. A parametric branch and bound method is developed which permits the solution of the knapsack problem for a range of values of the budget ceiling.

Two formulations are proposed for a special case of deferred capital investments, referred to as the multi-knapsack problem. The first formulation, after a transformation by means of a model equivalent, leads to a branch and bound algorithm which requires the solution of a standard transportation problem with surplus and deficits and certain routes prohibited at each step of the algorithm. The second model, although it may require a larger tree before optimality is reached, permits the solution by inspection of the linear program associated with each node of the solution tree.

The final part of this thesis studies capital investment for dependent proposals in the context of urban transportation planning. The branch and bound algorithm is adapted to the link addition network design problem, where a descriptive traffic assignment model is employed.

Finally, for the multistage link-addition network synthesis problem, a normative model is formulated as a block-angular mixed-integer linear program. A partitioning technique is employed to take advantage of the highly-structured form of the model.

We conclude with a detailed presentation of the partitioning technique of Benders, as applied to both continuous and mixed-integer programming problems presenting a block-angular structure.

ACKNOWLEDGEMENT

This report is essentially a reproduction of the thesis which was submitted by the author in January 1968 in partial fulfillment of the requirements for the degree of Doctor of Science.

The author wishes to thank the Banco de Mexico, S. A. for their financial assistance during his first two years at M.I.T., and to the Club Rotario de la Ciudad de México for their financial support during the academic year 1964-1965.

This report is part of a series in a program of research into the process of transportation systems planning. Additional support for preparation and reproduction was provided by the U.S. Department of Transportation, Office of Systems Analysis, through contract 7-35140 (DSR 70386), the General Motors Grant for Highway Transportation Research (DSR 70065); and by the Ford Foundation through a grant to the Urban Systems Laboratory of M.I.T. The support of these agencies is gratefully acknowledged.

To Professor Alan Hershdorfer, his thesis supervisor, he is particularly indebted for initiating and stimulating his interest in the research and teaching of optimization theory.

The author is grateful to Professor C. L. Miller, Head of the M.I.T. Department of Civil Engineering, and Professor Daniel Roos, Director of the Civil Engineering Systems Laboratory, for allowing him to devote a great portion of his time to this thesis during the past year.

A very special thank you goes to his friend and colleague David K. Bivins for his valuable comments and his excellent job in editing this thesis.

Thanks to Mrs. Linda Frechette's most accurate and enthusiastic typing this work was produced.

A very heartfelt thank you is extended to the author's family and his wife's for their constant encouragement during his graduate work at M.I.T.

To his wife Sandy, he dedicates this work in deepest gratitude for her continuing support and optimism in helping to make it a reality.

CONTENTS

CHAPTER I

FUNDAMENTAL CONCEPTS OF OPTIMIZATION THEORY

1.1 INTRODUCTION

The goal of this chapter is to formalize the concepts relevant to describing the nature and scope of *optimization theory*. We begin by defining the optimization problem and discussing its complex nature. We identify the fundamental steps in the solution process of optimization problems as: i) problem definition, ii) formulation of an optimization model, iii) selection of a solution method, and iv) application of the solution method. Each step of the process and its implications is discussed in detail for a variety of applications.

A classification of optimization models and of solution methods is presented. The material covered in this chapter and a historical survey of optimization theory (cf. Appendix B) are intended to present a general framework of the theory which will be applied in the main body of this work to specific types of optimization problems. Finally, we shall discuss some important aspects of optimization in the context of analysis and design of engineering systems.

1.2 THE OPTIMIZATION PROBLEM

Whenever an engineer or decision maker is confronted with the problem of selecting a course of action from a set of alternatives he will be compelled to choose, from the available alternatives, the best in terms of a certain predetermined goal or set of goals relevant to the nature of the problem.

It is assumed that the degree to which the goal or objective of the problem is reached for each alternative course of action can be evaluated by a quantitative method. In other words, a measure of the utility of each course of action may be obtained, allowing the decision maker to select the alternative yielding the maximum utility. The degree to which the goal is obtained is the *figure of merit* for a particular solution.

DEFINITION. An optimization problem is defined as the one of selecting among a set of various alternatives (possibly infinite) of a certain problem, the one for which a given figure of merit is optimized (i.e., maximized or minimized).

1.3 OPTIMIZATION THEORY. THE NATURE OF THE OPTIMIZATION PROBLEM

The nature of optimization problems is often quite complex, and a wide variety of cases presenting different characteristics is encountered in practical problems. To visualize the complexity which may be present in the nature of the problem, consider the following examples: i) a decision maker may be confronted with a problem having a clearly-defined objective to optimize; however, the problem may or may not be subject to a set of constraints. He may also have to consider the solution to the problem on the assumption of either deterministic or stochastic behavior. ii) the decision maker may have to interact and compete with other participants, each of whom is attempting to make decisions which optimize his own figure of merit. iii) several decisions may have to be made on a multistage problem, where the goal sought is a long-range optimization as opposed to suboptimization of a particular stage of the problem.

It is this complex nature as well as the different structural characteristics of the models (cf. Section 1.6) that clearly indicate the need for a variety of techniques to cope with the solution of optimization problems. The set of all these techniques, namely those included under the specific names of *mathematical programming, game theory, statistical decision theory, dynamic programming, control theory, calculus of variations*, etc., constitute with their theoretical foundations the general theory of optimization.

Optimization theory in its widest sense is the unified branch of *mathematical analysis* that provides a formal approach to the solution of optimization problems.

1.4 SOLUTION PROCESS

The solution process for optimization problems may not be identical in all cases and may differ depending on the special nature of the problem; nonetheless it will always be possible to distinguish in the process the basic steps indicated in Fig. 1-1. The various loops indicate possible revision of the previous decision.

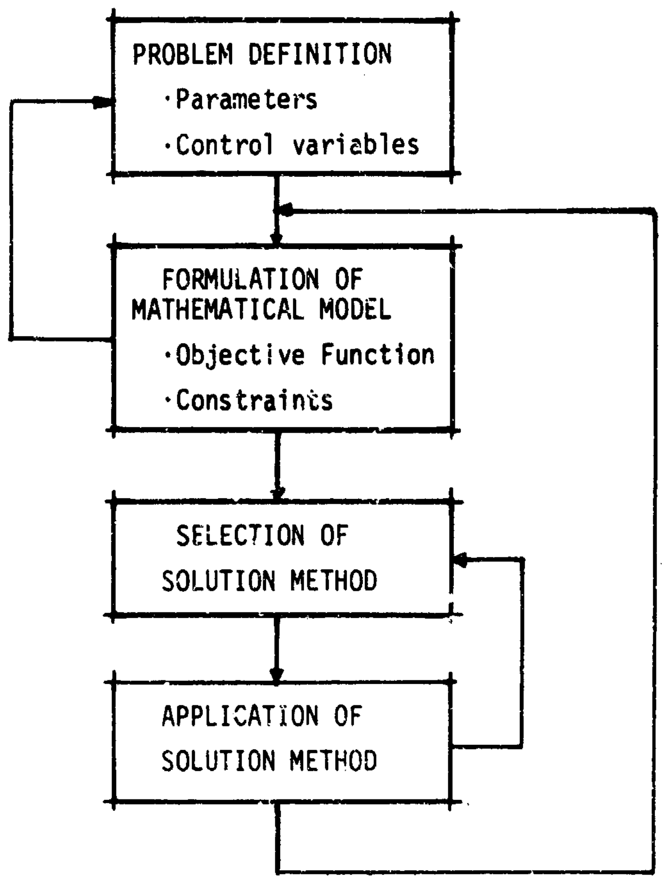


Fig. 1-1. Optimization Problem Solution Process

1.5 PROBLEM DEFINITION

At the problem definition stage the decision or control variables governing the problem are identified, and the form of interactions among the variables is specified. A figure of merit must be defined in terms of the relevant control variables and the range of variation of the controls must be explicitly or implicitly specified. Finally, the constraints to be satisfied by the variables must also be established.

1.6 FORMULATION OF A MATHEMATICAL MODEL

Once the problem has been properly defined, the subsequent step will be to formulate an abstract model (usually a mathematical model), that faithfully represents the essential structure of the problem and that may be amenable to solution through application of a well-known procedure. Whenever reference is made to models it will be understood in the sense of Karlin*, "a model is a suitable abstraction of reality preserving the essential structure of the problem in such a way that its analysis affords insight into both the original concrete situation and other situations which have the same formal structure".

It is clear that solution of the model will produce accurate results only to the extent that the model is representative of the original problem. If the problem has not been properly modeled, its solution may lead to dubious results or completely erroneous ones; for instance, consider the case of a linear programming model giving an unbounded solution as a result of a constraint of the problem not being included in the model.

We shall now analyze some distinctive characteristics of optimization models that will permit their convenient classification. This will be useful for further identification of the models that will be encountered in subsequent chapters.

We shall distinguish three main components of an optimization model: i) the set of problem variables, ii) the figure of merit to be optimized, iii) the domain of definition of the problem variables (determined by the constraints of the problem). The optimal solution for certain classes of optimization problems consists of numerical values taken by the problem variables, satisfying the constraints and simultaneously optimizing the figure of merit. Other classes of optimization problems seek to find a

*Karlin, S., *Mathematical Methods and Theory in Games, Programming, and Economics*, Vol. I, Addison-Wesley, 1959, p. 1.

curve or function (variational problems), that satisfies a set of constraints and renders optimal a certain functional expression of the set of feasible solution curves.

For certain problems the objective will be amenable to a closed form mathematical representation as a function of the control variables. For other problems this closed representation might not be obtainable, and the figure of merit for a given set of values of the control variables may only be known after a complex process has been completed (such as a simulation process, an engineering analysis, the solution of an elaborate computer program, or a table look-up).

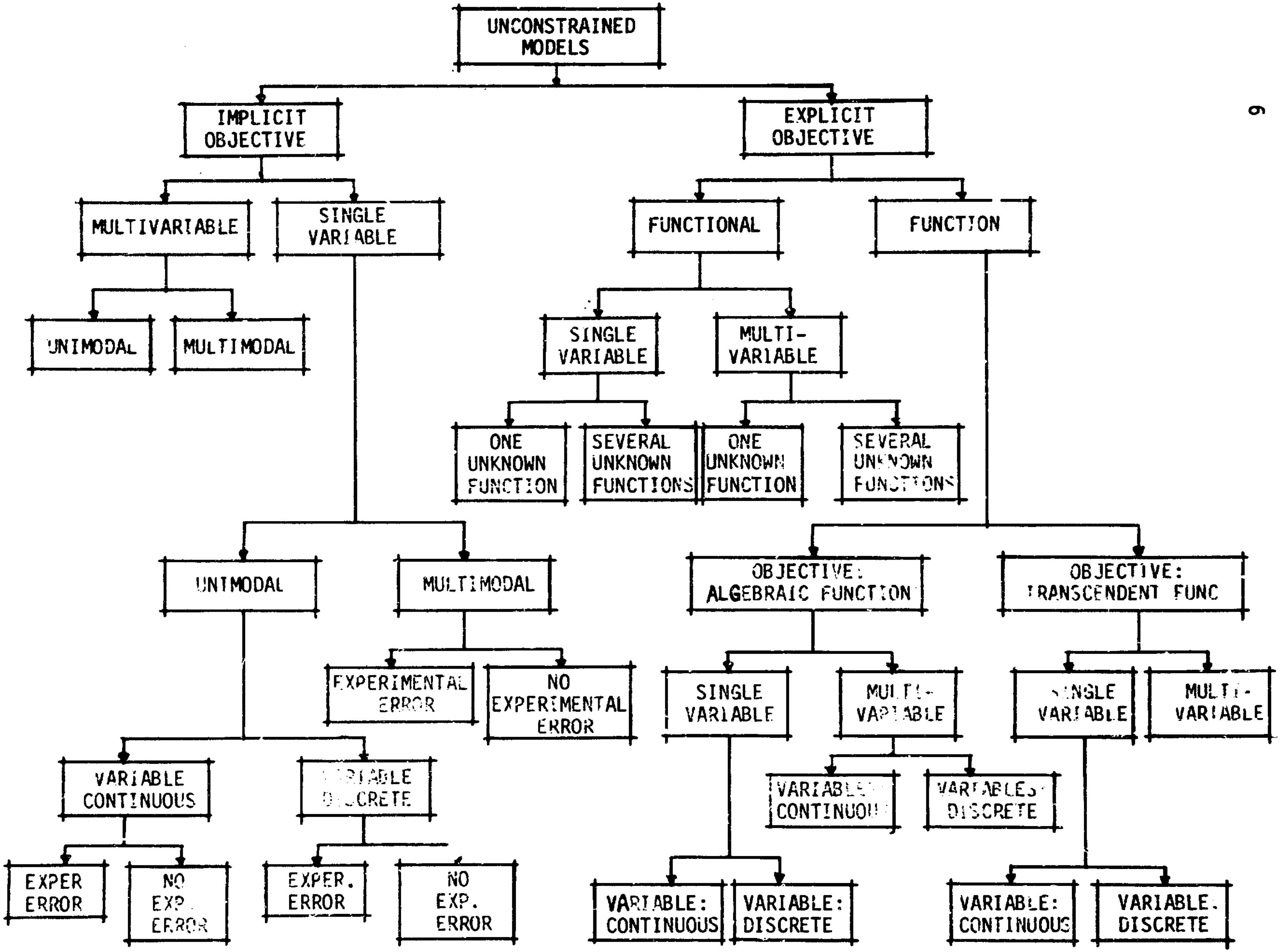
Furthermore, the problem may be constrained or unconstrained. For constrained problems capable of formulation in a closed form mathematical representation, the nature of the constraint expressions may be quite diverse. For instance they may be algebraic or transcendent expressions, equalities or inequalities, linear or nonlinear with the domain of the variables being a discrete set or the continuum. Also some of the constraints may be differential equations or definite integrals.

In the light of the above discussion we have developed the tree-structured classification of optimization models illustrated in Fig. 1-2. The tree obviously may be expanded in both the vertical and horizontal directions to make it as complete as is needed or desired.

We shall be able to distinguish certain branches of the tree, representing specific classes of problems, for which the solution procedures form a well-established mathematical development. For instance, models in the constrained optimization branch for which both the constraints and objective may be represented in closed algebraic form constitute that part of optimization theory generally known as *mathematical programming*.

As a second example, consider the class of problems for which the explicit objective function is expressed by a definite integral (functional objective) with or without subsidiary conditions. The solution of such models falls within the scope of the *classical calculus of variations*.

Finally, consider those models with constraints and/or objective lacking a closed mathematical representation. The optimization of such models must be attained by any means short of brute force; the techniques usually applied fall under the general name of *direct search methods*.


UNCONSTRAINED MODELS
IMPLICIT OBJECTIVE
EXPLICIT OBJECTIVE
MULTIVARIABLE
SINGLE VARIABLE
FUNCTIONAL
FUNCTION
UNIMODAL
MULTIMODAL
SINGLE VARIABLE
MULTI-VARIABLE
ONE UNKNOWN FUNCTION
SEVERAL UNKNOWN FUNCTIONS
ONE UNKNOWN FUNCTION
SEVERAL UNKNOWN FUNCTIONS
UNIMODAL
MULTIMODAL
OBJECTIVE: ALGEBRAIC FUNCTION
OBJECTIVE: TRANSCENDENT FUNC
EXPERIMENTAL ERROR
NO EXPERIMENTAL ERROR
SINGLE VARIABLE
MULTI-VARIABLE
SINGLE VARIABLE
MULTI-VARIABLE
VARIABLE CONTINUOUS
VARIABLE DISCRETE
VARIABLES CONTINUOUS
VARIABLES DISCRETE
EXPER ERROR
NO EXP. ERROR
EXPER. ERROR
NO EXP. ERROR
VARIABLE: CONTINUOUS
VARIABLE: DISCRETE
VARIABLE: CONTINUOUS
VARIABLE. DISCRETE

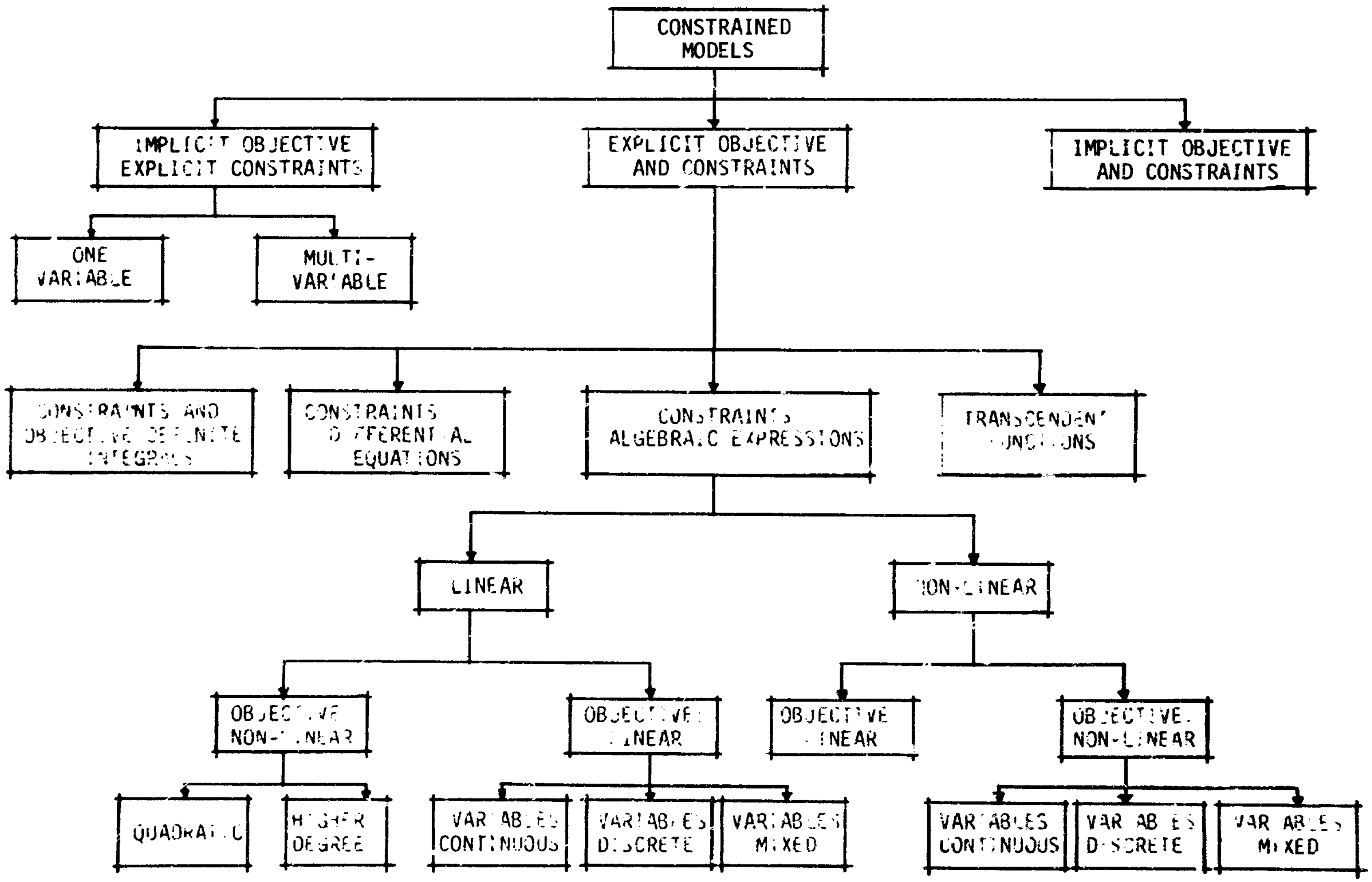


FIG 1-2 Classification of Optimization Models

An example of this class would be a certain stochastic process (e.g. a waiting line, a given renewal process) being analyzed by means of a costly computer simulation. The input parameters may be varied and the simulation executed for each set of values. Associated with the output of each run, a *measure of effectiveness* (MOE), of the corresponding input parameters may be estimated. If the problem is to select the input parameters that optimize the MOE, a direct search technique is required in this case to find the optimum while minimizing the number of simulated trials.

1.7 SOLUTION TECHNIQUES

Solution techniques are the procedures and algorithms devised for the solution of optimization problems. The actual solution usually entails determination of numerical values of the control variables and the optimum value of the figure of merit.

Optimization methods are usually broken down into two major categories: *indirect and direct methods*. With direct methods, the optimum solution is sought by directly calculating values of the objective function at different points of the feasible domain. The values thus obtained are compared and, by means of an auxiliary criterion, a new point is next analyzed which hopefully will improve the value of the objective function.

Alternatively, indirect methods look for a set of values of the control variables that satisfy known necessary conditions for optimality. The classical method of the differential calculus is an example of the indirect type. In effect, values of the variables are sought for which the first derivatives of the objective function vanish, provided that continuity of the function and existence of derivatives in the region of interest are guaranteed. In this way, the optimization problem has been transformed into a root-finding problem.

The Simplex algorithm of linear programming exhibits features of both the direct and indirect methods. It performs a direct search over extreme points of the feasible domain only (points satisfying the necessary condition for an optimum) in such a way that the objective function is at least as good as in the previous step. Finally, the optimum among the set of extreme points is detected when the indirect criterion of feasibility of the complementary solution to the associated dual problem is satisfied.

For certain mathematical models of optimization, a solution method may include transforming the original model into an equivalent one that promises to be more tractable than the former (cf. Chapters III and IV). Consider the methodology of *geometric programming**; in this case, the polynomial optimization is formulated in terms of its dual problem and this is the model that is actually solved. Another example is the transformation into a linear programming problem of a *separable* nonlinear program.

Direct techniques may be subdivided into two major groups: *simultaneous* and *sequential* methods. Simultaneous search techniques calculate values of the objective function or response surface at a set of points determined *a priori* by a certain search strategy. Sequential search methods, on the other hand, deal with sequential examination of trial solutions, basing the location of subsequent trials on the results of earlier ones. We present in Fig. 1-3 a subset of representative solution techniques for each one of the classes of methods discussed in this section.

1.8 SELECTION OF A METHOD

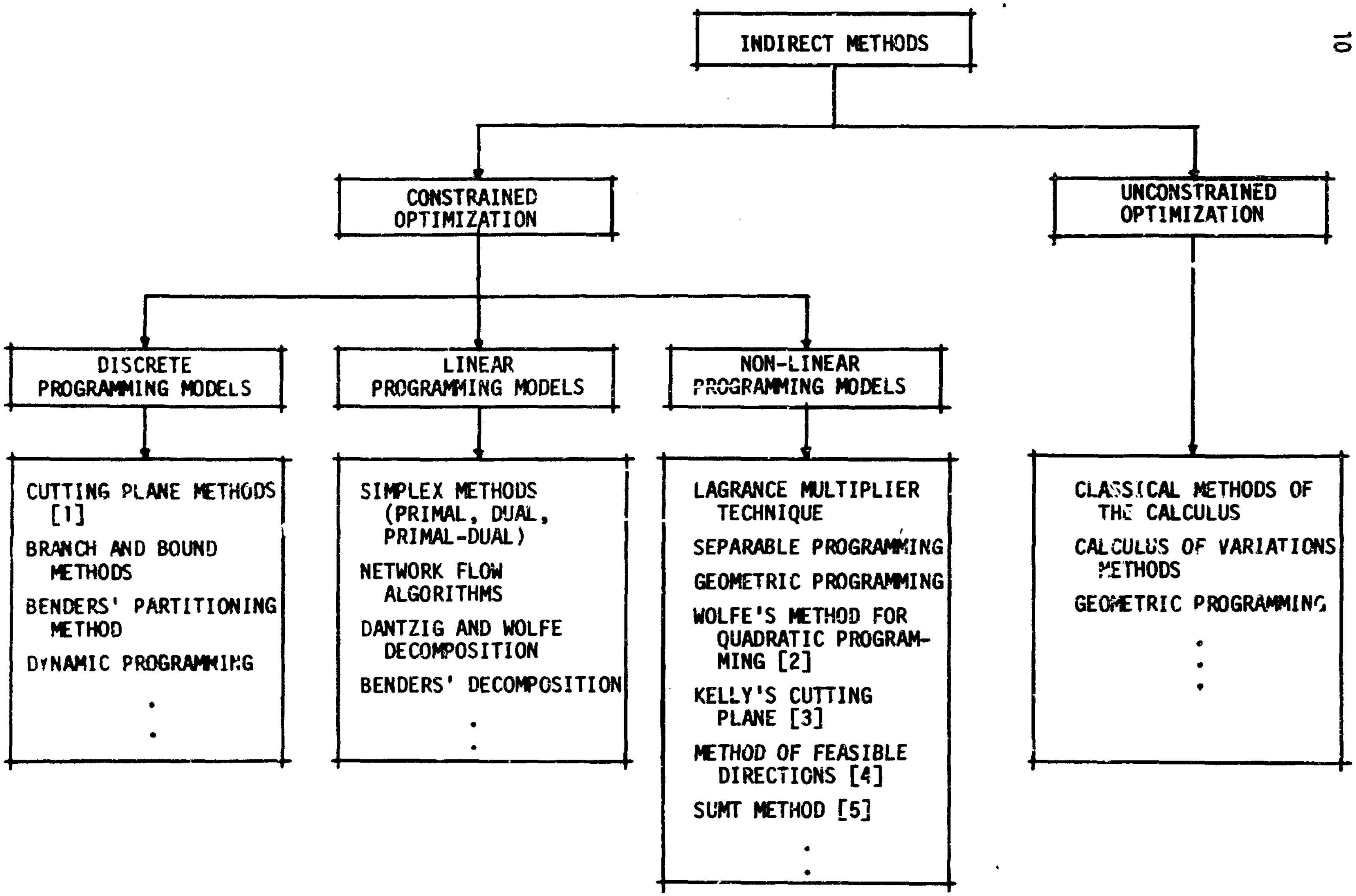
The selection of a convenient solution method for a given problem depends on the type of model employed, the existing solution techniques for that particular model, and the computation facilities available to the engineer-analyst.

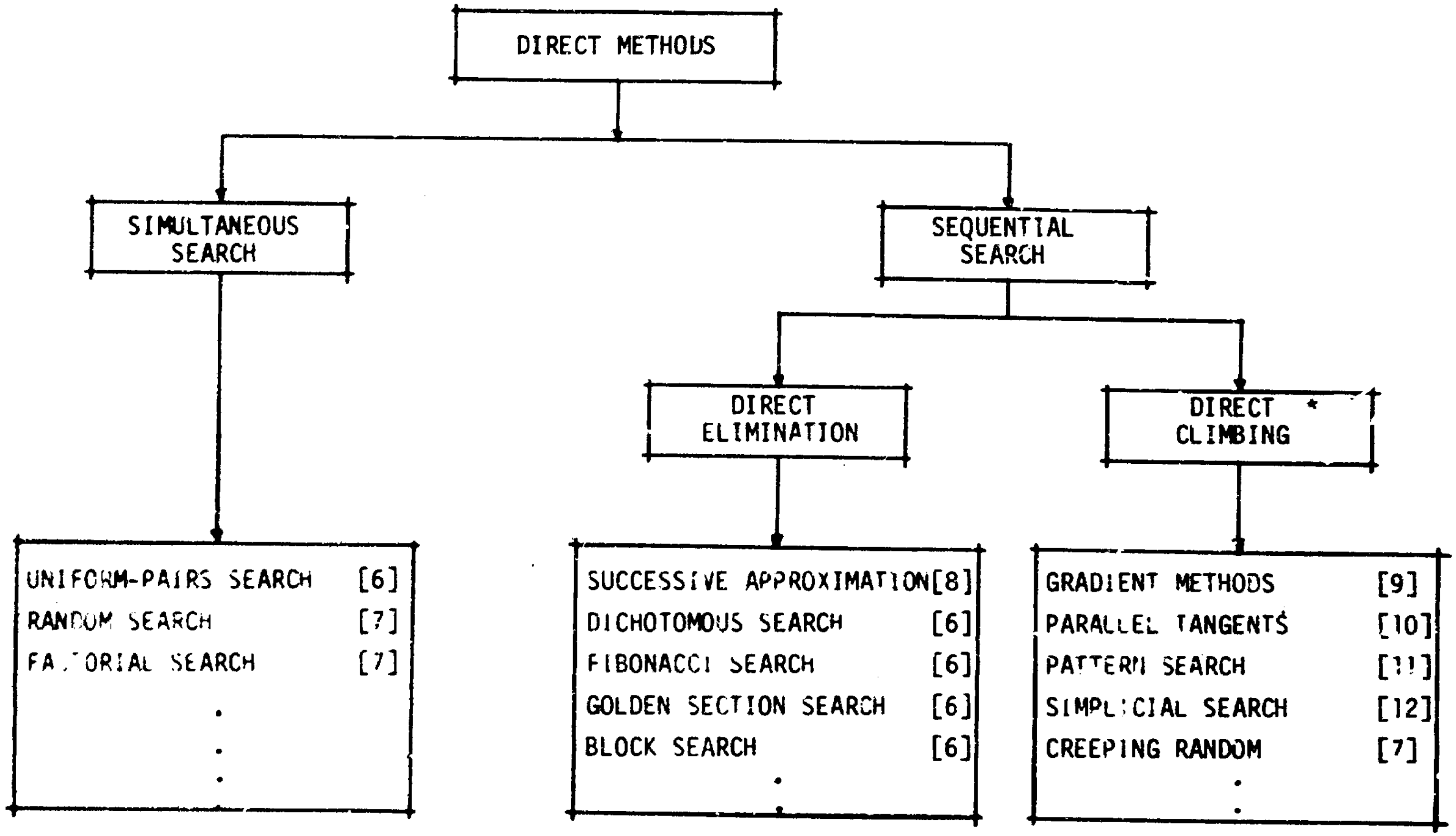
In the selection process one may consider such factors as linearities of the model, number of variables, number of constraints, special structures, separability or weak-coupling of variables in constraints and/or objective, objective or constraint surfaces of readily interpreted geometric character, etc.

The final selection of a well-suited method for a particular problem depends then on the detailed properties of the model as well as the solution techniques that form part of a software package of an available computer installation.

---

* Duffin, R. J., E. L. Peterson, and C. M. Zener, *Geometric Programming*, John Wiley, 1967


INDIRECT METHODS
CONSTRAINED OPTIMIZATION
UNCONSTRAINED OPTIMIZATION
DISCRETE PROGRAMMING MODELS
LINEAR PROGRAMMING MODELS
NON-LINEAR PROGRAMMING MODELS
CUTTING PLANE METHODS [1]
BRANCH AND BOUND METHODS
BENDERS' PARTITIONING METHOD
DYNAMIC PROGRAMMING
SIMPLEX METHODS (PRIMAL, DUAL, PRIMAL-DUAL)
NETWORK FLOW ALGORITHMS
DANTZIG AND WOLFE DECOMPOSITION
BENDERS' DECOMPOSITION
LAGRANGE MULTIPLIER TECHNIQUE
SEPARABLE PROGRAMMING
GEOMETRIC PROGRAMMING
WOLFE'S METHOD FOR QUADRATIC PROGRAMMING [2]
KELLY'S CUTTING PLANE [3]
METHOD OF FEASIBLE DIRECTIONS [4]
SUMT METHOD [5]
CLASSICAL METHODS OF THE CALCULUS
CALCULUS OF VARIATIONS METHODS
GEOMETRIC PROGRAMMING



NOTE: The numbers in brackets correspond to bibliographical references in section 1.10

* For a discussion of elimination vs. climbing, see [6].

FIG. 1-3 Classification of Solution Methods

We have just presented a brief review of some classes of optimization problems, the mathematical models applicable to these problems, and the methods available for their solution. To place the developments of optimization theory in proper perspective, the reader is referred to Appendix B for a survey of the most significant contributions of different mathematicians through the centuries.

In the following section we shall discuss concepts relevant to engineering systems optimization, as a framework for the class of problems undertaken in the main body of the text.

1.9 ENGINEERING SYSTEMS OPTIMIZATION

The engineer uses analytical and experimental methods to analyze and interpret the behavior of the physical world in such a way that appropriate decisions can be made regarding investment of scarce resources for the development of facilities of economic utility.

In general, the engineer seeks a design which satisfies a certain specified performance of the facility in an economical manner. The meaning of *economical* is subject to various interpretations. It may mean a least cost design including both construction and operating costs. On the other hand, one may seek a design yielding the highest level of performance consistent with the given construction and operating budgets; one may also mix these extreme cases.

With this in mind, we can view the task of the engineer as that of providing the best solution to the problem as described; therefore, the engineer confronts an optimization problem in the sense discussed in previous sections.

From the practical, computational point of view, the majority of engineering system design problems are sufficiently complex that one cannot provide a mathematical model for the entire problem which could be solved by one of the solution techniques indicated previously.

However, any engineering system design problem is defined in terms of a set of boundaries which delineate the range of the systems of interest. These boundaries represent an arbitrary but presumably reasonable separation of the system under consideration from other systems in which it is imbedded. Hence, the design problem can be viewed as a suboptimization of a set of subsystems, the union of which compose the system of interest.

Therefore, it seems perfectly natural to fragment an engineering system design problem into components, some of which may be sufficiently limited as to permit the application of optimization techniques. It is evident that the set of optimum solutions to the selected components will not in general constitute an optimum to the original system but simply a suboptimal solution.

The traditional process for solving an engineering system design problem usually takes the form of a trial and error procedure. However, in those cases where systems design problems are successfully subjected to mathematical optimization techniques, the engineer-analyst draws boundaries about the fragment of the design problem so that a closed form mathematical representation of the system is obtained. Known optimization techniques are then applied to this representation or model, and an optimal solution to the design problem is calculated.

When it is possible to isolate a system fragment of significant physical extensiveness and calculate its optimum design by a convergent process, we say that a synthesis algorithm exists for the design of the system.

While it may not be possible to isolate a section of a design problem such that its optimization may be termed a synthesis procedure, one expects to find parts of engineering design problems whose solutions will be small-scale optimizations. The solution of these small-scale optimizations which occur as parts of the total system will be of special interest in the incremental process of developing a total synthesis algorithm.

Throughout this work we shall be concerned with exploring parts of engineering design problems, the solution of which may be solved by known optimization techniques. The first part of the material covers optimal allocation of capital resources to a finite set of facilities. Problems involving synthesis of transportation networks will be developed in the remaining parts of the work.

1.10 NOTES TO CHAPTER I

[1] Balinski, M. L., "Integer Programming: Method, Uses, Computation", *Management Science*, Vol. 12, No. 3, November 1965, pp. 255-264.

[2] Wolfe, P. "The Simplex Method for Quadratic Programming", *Econometrica*, Vol. 27, 1959, pp. 382-398.

[3] Kelley, J. E., Jr., "The Cutting Plane Method for Solving Convex Programs", *J. SIAM*, Vol. 8, December 1960, pp. 703-712.

[4] Zoutendijk, G., *Methods of Feasible Directions*, Elsevier Publishing Co., Amsterdam, 1960.

[5] Fiacco, A. V., and G. P. McCormick, "Computational Algorithm for the Sequential Unconstrained Minimization Technique for Nonlinear Programming", *Management Science*, Vol. 10, 1964, pp. 601-617.

[6] Wilde, D. J. and C. S. Beightler, *Foundations of Optimization*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1967.

[7] Brooks, S. H., "A Discussion of Random Methods for Seeking Maxima", *Operations Research*, Vol. 6, 1958, pp. 244-251.

[8] Berman, G., "Minimization by Successive Approximation", *J. SIAM Num. Anal.*, Vol. 3, 1966, pp. 123-133.

[9] Hadley, G., *Non Linear and Dynamic Programming*, Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1964, Ch. 9.

[10] Shah, B. V., R. J. Buehler, and O. Kempthorne, "Some Algorithms for Minimizing a Function of Several Variables", *J. SIAM*, Vol. 12, March 1964, pp. 74-92.

[11] Hooke, R. and T. A. Jeeves, "'Direct Search' Solution of Numerical and Statistical Problems", *J. Assn. Comp. Mach.*, Vol. 8, April 1961, pp. 212-229.

[12] Spendley, N., G. R. Hext, and F. R. Himsworth, "Sequential Application of Simplex Design in Optimization and Evolutionary Operations", *Technometrics*, Vol. 4, November, 1962, pp. 441-459.

CHAPTER II

A BRANCH AND BOUND ALGORITHM FOR A CLASS OF DISCRETE OPTIMIZATION PROBLEMS

2.1 INTRODUCTORY REMARKS

Branch and bound algorithms, a class of solution methods for integer programming problems, have been extensively studied since the first procedure of this class, offering a new and fresh approach to the solution of combinatorial problems, was published by Land and Doig [1] in 1960.

The name *branch and bound* is due to Little et al. [2]. These authors successfuly employed a technique in this class to obtain a solution to the traveling saleman problem which was substantially more efficient than solutions previously available. This result encouraged further investigation into the applicability of this technique. Improvements of existing methods were carried out by Dakin [3] and Driebeek [4], and further applications are due to Ignall and Schrage [5] on the job scheduling problem, to Efroymson and Ray [6] on a plant location problem, Hershdorfer et al. [12] on the assignment of numbers to nodes of a tree-dimensional grid so that the bandwidth of the associated node-node incidence matrix is minimized, and to Gavett and Plyter [7] on the optimal assignment of facilities to locations. A survey on the state of the art up to 1966 may be found in the work of Lawler and Wood [9].

Various formalizations of the general class of branch and bound methods have been undertaken by Agin [8], Lawler and Wood [9], Roy, Nghiem, and Bertier [10] and others. Most recently Ichbiah [11] generalized the work of Roy, et al. and developed a parametric branch and bound technique.

In subsequent chapters we shall study various optimization problems arising in the fields of transportation systems analysis and design, and capital budgeting for independent and dependent projects. These problems will be mathematically formulated as discrete-bivalent programming problems (i.e., one in which a pair of feasible values is specified for each member of a subset of problem variables). These problems arise in apparently independent areas but it is possible to develop mathematical models for these problems, which in fact are closely related. These models can all be solved by a branch and bound technique of the Land and Doig type requiring the solution of network flow or transportation type problems at each step of the iterative procedure.

Since each of the problems we shall consider is solved by a variant of our branch and bound technique, this chapter presents the general formulation of this method as a basis for the particular applications.

The problems that we shall study share the characteristic that a feasible solution can be obtained with little or no computational effort at every stage of the algorithm. Associated with this property is a means for developing both an upper and a lower bound to the objective function at each stage of the procedure. This double bounding technique leads to a reduction of the search space and to an increase in the efficiency of the solution technique.

The next section describes the mathematical structure associated with our class of problems, and subsequent sections describe the common elements of the solution technique and prove its validity and finite convergence.

2.2 MATHEMATICAL FORMULATION FOR THE CLASS OF PROBLEMS

Let $\underline{x}$ denote a vector in E^n and S_1 a closed and bounded convex set with boundaries defined by hyperplanes in E^n. Let T_1 be a finite nonempty set of vectors in the same space, and denote by Ω_1 a finite subset of S_1 obtained by the intersection of S_1 and T_1, $\Omega_1 = S_1 \cap T_1$.

Consider the following discrete optimization problem the solution of which is to be obtained.

$$
\begin{array}{lll}
P : & \text{Determine} & \underline{x}^0 \text{ and } z^0 \text{ so as to} \\
 & \text{Minimize} & z = f(\underline{x}) \\
 & \text{Subject to} & \underline{x} \in \Omega_1
\end{array}
$$

where f is a single-valued function of $\underline{x}$.

DEFINITION. Let us denote by A_j the j'th auxiliary continuous problem, derived from P as follows:

A_j : Determine $\underline{x}^*(j)$ and $z^*(j)$ so as to
Minimize $z(j) = f(\underline{x})$
Subject to $\underline{x} \in S_j$, $j = 1,2,\ldots$

For $j=1$, S_1 is given and for $j>1$, S_j is a subset of S_1 to be defined in section 2.4. We shall assume that a finite algorithm, to be called after Dakin [3], the *sub-algorithm*, exists for solution of problem A_j. Furthermore, it is assumed that a feasible solution to problem P may be determined, for each j, by "simple inspection" of the solution to problem A_j. This solution will be denoted by $\underline{\hat{x}}(j)$, $\hat{z}(j)$. The "inspection" to be performed on the optimum solution to A_j to obtain a feasible solution to P will be called a *rounding operation.*

2.3 THE DIRECTED TREE

The branch and bound algorithm for solution of problem P, to be set forth in section 2.5, is an iterative technique that may be interpreted as the generation of a directed tree: $T(i) = [N(i), A(i)]$, where $N(i)$ and $A(i)$ are respectively the set of nodes and the set of directed arcs at the end of iteration i. At each iteration, except for the first one during which only the root node of the tree is created, two new directed arcs and nodes will be added to the sets N and A.

Associated with each node $j \in N(i)$ are a subset Ω_j of Ω_1 and a subset S_j of S_1, and associated with each arc $(j,k) \in A(i)$ is a set V_{jk} (cf. Section 2.4).

At the end of the i'th iteration, the subset of N(i) corresponding to the terminal nodes of the tree will be denoted by C(i). The set C(i) will be partitioned further into three subsets, F(i), E(i) and R(i) such that $F \cup E \cup R = C$. Set F will be called the set of *feasible or active nodes*, E the set of *infeasible or excluded nodes*, and R the set of *rejected nodes*.

The algorithm starts by generating the root of the tree, node 1, associating S_1 to it and solving A_1. From then on, and in an iterative fashion, bifurcating arcs and their corresponding nodes are added to the tree according to a *branching operation*. These directed arcs have as origin a conveniently selected node from F(i). For each node j thus created, the values $\underline{x}^*(j)$, $z^*(j)$ and $\hat{\underline{x}}(j)$, $\hat{z}(j)$ are obtained by solving the auxiliary problem A_j and applying a *rounding operation* to its optimal solution.

This iterative procedure terminates when the solution to the original problem P, or sufficient evidence of the existence of no solution, has been obtained. This evidence is given by the operations of *bounding*, *exclusion* and *rejection* (to be defined), in conjunction with the branching and rounding operations mentioned above.

2.4 BRANCH AND BOUND OPERATIONS

DEFINITION 1. Branching Operation. Let Ω_j, a non-empty subset of Ω_1 (if $\Omega_j = \Phi$, no branching operation will take place, see Definition 3), and $S_j \subset S_1$ be the sets associated with node j. The branching operation is defined by a partition of Ω_j into two subsets Ω_r and Ω_{r+1} such that:

$$\Omega_r \cup \Omega_{r+1} = \Omega_j \tag{2.1}$$

$$\Omega_r \cap \Omega_{r+1} = \Phi \tag{2.2}$$

where Φ is the empty set. This partition is achieved by creating two directed arcs (j,r) and (j,r+1) emanating from node j with associated sets $V_{j,r}$ and $V_{j,r+1}$ and two nodes r and r+1 with associated sets Ω_r and Ω_{r+1} such that:

$$\begin{aligned} \Omega_j \cap V_{j,r} &= \Omega_r \\ \Omega_j \cap V_{j,r+1} &= \Omega_{r+1} \end{aligned} \tag{2.3}$$

We observe the following theorem which characterizes $V_{j,r}$ and $V_{j,r+1}$:

THEOREM 2.1 *Given Ω_j, sufficient conditions for $V_{j,r}$ and $V_{j,r+1}$ to define a partitioning satisfying (2.1) and (2.2) are:*

$$V_{j,r} \cap V_{j,r+1} = \Phi \tag{2.4}$$

$$\Omega_j \subseteq (V_{j,r} \cup V_{j,r+1}) \tag{2.5}$$

Proof: Assume that (2.4) and (2.5) hold; then by intersecting both sides of (2.4) with Ω_j we obtain

$$\Omega_j \cap (V_{j,r} \cap V_{j,r+1}) = \Omega_j \cap \Phi$$

and since the intersection of sets is distributive

$$(\Omega_j \cap V_{j,r}) \cap (\Omega_j \cap V_{j,r+1}) = \Phi$$

Hence from (2.3) $\quad \Omega_r \cap \Omega_{r+1} = \Phi$

Finally (2.5) is equivalent to

$$\Omega_j \cap (V_{j,r} \cup V_{j,r+1}) = \Omega_j$$

$$\text{or} \quad (\Omega_j \cap V_{j,r}) \cup (\Omega_j \cap V_{j,r+1}) = \Omega_j$$

Hence, from (2.3) $\Omega_r \cup \Omega_{r+1} = \Omega_j$. This completes the proof.

Next we associate to nodes r and r+1 the subsets S_r and S_{r+1}, defined as follows:

$$\begin{aligned} S_r &= S_j \cap V_{j,r} \\ S_{r+1} &= S_j \cap V_{j,r+1} \end{aligned} \tag{2.6}$$

From the results of lemma 2.1 below, Ω_j is a subset of S_j. We observe that only the following condition is satisfied for S_r and S_{r+1}:

$$S_r \cap S_{r+1} = \Phi \tag{2.7}$$

That is, the sets S_r and S_{r+1} are mutually exclusive although they may not be collectively exhaustive of S_j.

After a finite number of branching operations have been performed, we may expect to have generated nodes t for which Ω_t has been reduced to a single element of the original domain Ω_1. It is also expected that the corresponding S_t is reduced to contain exclusively the same single element, so that $\Omega_t \equiv S_t$. We observe that this is possible for Ω_t since, by hypothesis, Ω_1 is finite; but this is not so for S_t, since S_1 is infinite. Hence, in order to guarantee that eventually $\Omega_t \equiv S_t$ we have to restrict further the sets $V_{j,r}$ and $V_{j,r+1}$ in the following way: it is assumed that the sets $V_{j,r}$ and $V_{j,r+1}$ are such that in a finite number of branching operations, nodes with Ω_t containing one single element have an associated S_t containing only the same single element. In the case of the particular applications considered in the present work, this is a relatively simple condition to satisfy. Finally, to initialize and make possible the branching operation, the sets Ω_1 and S_1 are assigned to the root node of the solution tree.

LEMMA 2.1 *Let Ω_r and S_r be the sets associated with node r of $T(i)$. Then Ω_r is a subset of S_r.*

Proof: By induction. In effect, for $r = 1$, $\Omega_1 \subset S_1$ by hypothesis. Let us assume that for node j, $\Omega_j \subseteq S_j$ is satisfied. Then letting r be the immediate successor of j and by intersecting each side of (2.3) and (2.6) we have

$$\Omega_r \cap S_r = (\Omega_j \cap S_j) \cap V_{j,r}$$

But from the previous assumption, $\Omega_j \cap S_j = \Omega_j$ and therefore

$$\Omega_r \cap S_r = \Omega_j \cap V_{j,r}$$

Or using (2.3), $\Omega_r \cap S_r = \Omega_r$, or equivalently $\Omega_r \subseteq S_r$, which completes the proof.

Note that for a terminal node $j=t$ with S_t containing a single element $\underline{x}$ such that $\underline{x} \in \Omega_1$, then $\Omega_t \equiv S_t$.

DEFINITION 2. Bounding operation. Given the current set $F(i)$ of active nodes, the bounding operation will be defined by means of the following actions:

a) Lower bounding operation. This consists of selecting the node $k \in F(i)$ such that

$$z^*(k) = \min_{j \in F(i)} \{z^*(j)\} \tag{2.8}$$

and of setting the value L_i, the current lower bound for problem P, equal to the value given by (2.8):

$$L_i = z^*(k) \tag{2.9}$$

Node k is said to be the *bounded node* for iteration i, defining the node from which branching will take place at the next iteration.

b) Upper bounding operation. This consists of finding the value

$$\hat{z}(s) = \min_{j \in F(i)} \{\hat{z}(j)\} \tag{2.10}$$

which constitutes the current least upper bound of the problem, and of setting the value U_i equal to the value given by (2.10):

$$U_i = \hat{z}(s) \tag{2.11}$$

Expressions (2.9) and (2.11) constitute, respectively, the best lower and upper bounds of the problem at the end of the i'th iteration. This statement will be substantiated by means of the following lemmas.

LEMMA 2.2 *If node j is the immediate predecessor of r then*

$$z^*(j) \leq z^*(r)$$

Proof: If $\underline{x}^*(j) \equiv \underline{x}^*(r)$ then $z^*(r) = z^*(j)$ since both A_r and A_j share the same objective function. Otherwise, if $\underline{x}^*(j) \neq \underline{x}^*(r)$ and since $S_r \subset S_j$ due to the way the branching was defined, problem A_r is more restricted, and consequently $z^*(j) \leq z^*(r)$.

LEMMA 2.3 *Let k be the bounded node of iteration i with associated value L_i given by (2.9). If $\underline{x}^0$, z^0 is the optimal solution to P, then $L_i \leq z^0$.*

Proof: Let L_i and L_{i+1} be the values given by (2.9), associated with any two consecutive iterations. From lemma 2.2 and since branching occurs from the last bounded node, it follows that $L_i \leq L_{i+1}$ and thus $L_i \leq L_\ell$, $\ell \geq i$. Now assume that the process of branching continues until the entire tree has been developed at iteration $\ell = t$. The set F(t) will contain all nodes associated with feasible solutions to P (guaranteed by branching operation). Then $L_t = \min_{j \in F(t)} [z^*(j)] = z^0$. Hence $L_i \leq L_t = z^0$, which completes the proof.

From lemma 2.2 and the definition of bounding, we observe that at each iteration the bounding operation indeed gives a lower bound to problem P as indicated by lemma 2.3; and also a better lower bound, (closer to the optimum) than the previous iteration as asserted by lemma 2.2 and the fact that branching occurs from the bounded node of the previous iteration.

LEMMA 2.4 *If $\hat{\underline{x}}(j)$, $\hat{z}(j)$ is the feasible solution to problem P obtained from a rounding operation at node j, then $z^0 \leq \hat{z}(j)$.*

Proof: If $\hat{\underline{x}}(j) \equiv \underline{x}^0$ it follows that $z^0 = \hat{z}(j)$. Otherwise $\hat{\underline{x}}(j) \neq \underline{x}^0$, and since $\hat{\underline{x}}(j)$ is only a feasible solution to P, then $z^0 \leq \hat{z}(j)$.

The rounding operation thus provides an upperbound on z^0 at each node it is performed upon. Now, since at the end of iteration i the best lower bound corresponding to node k is L_i, and the best upper bound corresponding to node s is U_i, the following theorem results:

THEOREM 2.2. *At any iteration,* $L_i \le U_i$ *and if* $L_i = U_i$*, the optimal solution has been obtained; it corresponds to the rounded solution of node* s *of the current iteration.*

Proof: The first part is evident: from the definition of bounding and from lemmas 2.3 and 2.4, it follows that $L_i \le z^0 \le U_i$ and therefore $L_i \le U_i$.

It remains to be proved that if $L_i = U_i$, node s is an optimum solution. Additional branching would make the lower bound greater than U_i; and since a feasible solution to P associated with node s has already been found, all other feasible solutions to P not yet discovered would yield no improvement in z^0 given by s. Consequently, the feasible solution to P associated with node s is the optimal.

DEFINITION 3. Exclusion. The exclusion operation is defined for a terminal node r of C(i) for which the corresponding set Ω_r is empty. Since Ω_r is empty, no need exists to consider further branching from node r and, as part of the exclusion operation, the node is assigned to the set E(i) of excluded nodes.

LEMMA 2.5 *If the solution to* A_r *is infeasible, then* $\Omega_r = \Phi$.

Proof: If A_r is infeasible, its domain of definition is empty: $S_r = \Phi$. From lemma 2.1, since $\Omega_r \subset S_r$, it follows that $\Omega_r = \Phi$.

DEFINITION 4. Rejection The rejection operation on node r consists of assigning the node to the set R(i) of rejected nodes if the following condition is satisfied:

$$z^*(r) > U_{i-1}$$

LEMMA 2.6 *If for node* r *at iteration* i, $z^*(r)$ *is greater than the upperbound at the previous iteration, no further branching from* r *is necessary.*

Proof: Consider the following possible cases:

i) $\underline{x}^*(r) \notin \Omega_1$ and $z^*(r) > U_{i-1}$: Since U_{i-1} is by definition an upperbound of the problem, it follows that Ω_r does not contain the optimum solution to P, and no further branching is required.

ii) $\underline{x}^*(r) \in \Omega_1$ and $z^*(r) > U_{i-1}$: Although node r is a feasible solution to P, the same argument as for case i) holds.

DEFINITION 5. Rounding operation. Let $\underline{x}^*(j)$, $z^*(j)$ be the solution to A_j associated with node j. The rounding operation consists of obtaining from $\underline{x}^*(j)$, $z^*(j)$ a feasible solution $\hat{\underline{x}}(j)$, $\hat{z}(j)$ to problem P.

For the classes of problems considered throughout this work, unless otherwise indicated, this operation is possible by conveniently rounding off certain components of $\underline{x}^*(j)$. When this operation is possible, the double bounding feature of the algorithm may be employed, thus resulting in an improved branch and bound method.

We note that if the operation is possible for each node j, then:

a) The upperbound U_i may be updated at each iteration, thus making possible the execution of the rejection operation. Since a rejected node is assigned to the subset R(i), and the selection for branching is performed among the nodes in subset F(i), no further information associated with the rejected node is required.

b) The updating of the upperbound U_i at each iteration reduces the interval of uncertainty of the optimal solution z^0 at each iteration, since $L_i \leq z^0 \leq U_i$. Furthermore, if the branch and bound method is used for suboptimization, and the process is terminated before an optimal solution has been obtained, the algorithm nonetheless provides valuable information at that step. In effect, the available information is represented by a feasible solution to the original problem P, plus a lower bound on the problem that permits us to estimate how far the available feasible solution is from optimality.

c) A measure of effectiveness for the rounding operation is provided by the algorithm. Note that the set C(i) of terminal nodes of iteration i contains exactly i nodes. This is true, since at each iteration two new nodes are created, r and $r+1$ and the node j from which branching occurred is no longer terminal, hence the net increase is one terminal node. Furthermore, according to the partition of $C(i)$ defined earlier, each terminal node is assigned to one of the sets $F(i)$, $E(i)$ or $R(i)$.

Thus, if we let α be number of elements in $F(i)$ and β the number of elements in $R(i)$, a measure of effectiveness (MOE) of the rounding operation may be defined as

$$\text{MOE} = \frac{\beta}{\alpha+\beta}$$

With the possible operations and associated lemmas established, we now proceed to describe the algorithm

2.5 SPECIFICATION OF THE ALGORITHM

The branch and bound algorithm consists of an initial step that generates the root of the directed tree (iteration 1), plus subsequent analogous iterations, continued until either the optimal solution or sufficient evidence of the existence of no solution is obtained. Note that under the assumption that the rounding operation is possible, P will always have a feasible solution.

STEP 1. Set $i=1$ and create node $j=1$. Set $F(1)=E(1)=R(1)=\Phi$. Solve A_1. If the solution is infeasible, stop; problem P has no solution. Otherwise, if $\underline{x}^*(1) \in \Omega_1$, stop: the solution is optimal. If $\underline{x}^*(1) \notin \Omega_1$, *bound* node 1 with $L_1 = z^*(1)$. *Round* node 1 to obtain $\hat{\underline{x}}(1)$, $\hat{z}(1)$. Set $U_1 = \hat{z}(1)$. If $L_1 = U_1$, stop; the rounded solution is optimal Otherwise, $L_1 < U_1$ Assign node 1 to $F(1)$, set $i = i+1$ and go to step i.

STEP i a) BRANCH Branch from bounded node $j \in F(i)$. Delete node J from $F(i)$ Create nodes r and $r+1$ and directed arcs (j,r) and $(j, r+1)$ Solve problems A_r and A_{r+1}, and in both cases do the following: if A_r (A_{r+1}) is infeasible, *exclude* node $r(r+1)$ by assigning it to $E(i)$ Otherwise A_r (A_{r+1}) has an optimum solution.

If $z^*(r)$ $(z^*(r+1)) > U_{i-1}$, *reject* node r(r+1) by assigning it to set R(i). Otherwise, $z^*(r)$ $(z^*(r+1)) \le U_{i-1}$, so assign node r(r+1) to set F(i).

b) ROUND. Round node r(r+1) if it was assigned to either F(i) or R(i).

c.1) BOUND FROM ABOVE. Set $U_i = \min [U_{i-1}, \hat{z}(r), \hat{z}(r+1)]$ for node $r(r+1) \varepsilon$ F(i) or R(i). Reject nodes of F(i) having $z^*(j) > U_i$ by assigning them to R(i).

c.2) BOUND FROM BELOW. Select node $k \varepsilon$ F(i) by using lower bound operation. Lower bound node k with $L_i = z^*(k)$. If $L_i = U_i$, stop; the feasible solution that provides the upperbound is optimal. Otherwise, $L_i < U_i$. Set i = i + 1 and go to step i.

It remains to be shown that the algorithm indeed finds the optimal solution in a finite number of steps. Since it is assumed that the rounding operation is possible, a feasible solution to P exists, and therefore an optimal solution exists. Moreover, from the way the branching operation has been defined and the hypothesis that Ω_1 is finite, the algorithm would, in a finite number of steps, generate all feasible solutions to P. (i.e., solutions corresponding to terminal nodes). And finally, from theorem 2.2, the optimal solution may be identified.

2.6 NOTES TO CHAPTER II

[1] Land, A. H., and A. Doig, "An Automatic Method of Solving Discrete Programming Problems", *Econometrica*, Vol. 28, 1960, pp. 497-520.

[2] Little. J. D. C., K. G. Murty, D. W. Sweeney, and C. Karel, "An Algorithm for the Traveling Salesman Problem", *Operations Research*, Vol. 11, 1963, pp. 972-989.

[3] Dakin, R. J., "A Tree-search Algorithm for Mixed Integer Programming Problems", *The Computer Journal*, Vol. 8, 1965, pp. 250-255.

[4] Driebeek, N. J., "An Algorithm for the Solution of Mixed Integer Programming Problems", *Management Science*, Vol. 12, No. 7, March 1966, pp. 576-587.

[5] Ignall, E. and L. Schrage, "Application of the Branch and Bound Technique to Some Flow-Shop Scheduling Problems', *Operations Research*, Vol. 13, 1965, pp. 400-412.

[6] Efroymson, M. A. and T. L. Ray, "A Branch-Bound Algorithm for Plant Location", *Operations Research*, Vol. 14, 1966, pp. 361-368.

[7] Gavett, J. W. and N. V. Plyter, "The Optimal Assignments of Facilities to Locations by Branch and Bound", *Operations Research*, Vol. 14, 1966 pp. 210-232.

[8] Agin, N., "Optimum Seeking with Branch and Bound", *Management Science*, Vol. 13, No. 4, December 1966, pp. B-176 - B-185.

[9] Lawler, E. L. and D. E. Wood, "Branch-and-Bound Methods: a Survey", *Operations Research*, Vol. 14, 1966, pp. 699-719.

[10] Roy, B., Nghiem, Ph. T. and P. Bertier, "Programmes Linéaires en Nombres Entiers et Procédure S.E.P.", *Metra*, 4, 1965, pp. 441-460.

[11] Ichbiah, J. D., *Connectivity Analysis and Branch and Bound Methods*, Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, Mass., August 1967.

[12] Hershdorfer, A. M., A. R. Raab, and G. M. Sturman, "On the Efficient Inversion of Large Structured Matrices", *Proceedings*, A.S.C.E., Engineering Mechanics Division Specialty Conference, Oct. 12-14, 1966, Washington, D. C., pp. 651-666.

[13] Lomnicki, Z. A., "A Branch and Bound Algorithm for the Exact Solution of the Three-machine Scheduling Problem", *Operational Research Quarterly*, Vol. 16, 1965, pp. 89-100.

[14] Balas, E., "An Additive Algorithm for Solving Linear Programs with Zero-One Variables", *Operations Research*, Vol. 13, 1965, pp. 517-546.

CHAPTER III

CAPITAL INVESTMENT ON INDEPENDENT PROJECTS

3.1 THE CAPITAL ALLOCATION PROBLEM

We shall refer to the problem of optimally allocating a fixed capital budget among a finite set of competing proposals as the *capital allocation problem**. We can make a basic distinction between two classes of allocation problems which will result in substantially different analytical formulations and hence different techniques to be used in their solution process. These correspond to the cases of *independent* and of *dependent* investment proposals. We shall consider as independent projects, after Lorie and Savage ([1], p. 229), those for which "the worth of individual investment proposals is not profoundly affected by the acceptance of others".

In this and in the following chapter we shall be concerned with optimal allocation of resources among independent proposals, while in subsequent chapters, optimal capital allocation for dependent projects will be studied for various problems in the context of transportation network synthesis.

Special cases of the capital allocation problem have been studied by Lorie and Savage [1] for the case of independent projects. They first consider the problem of allocating a fixed amount of money among competing alternatives, each requiring a given capital outlay in a single time period. The objective to be optimized is the sum of the net present values of the investments (i.e., the algebraic sum of positive and negative costs flows discounted to the present, using the firms "cost of capital" as the discount rate). Their proposed solution method is based on ranking the

*Although the discussion in this chapter is in terms of money allocation, it is in fact applicable to allocation of a variety of other scarce resources.

investment proposals in decreasing order of present value per dollar of outlay required, and accepting them in that order until the fixed budget is exhausted. They do not, however, deal with the ranking of various combinations of projects and therefore their method does not guarantee an optimal solution.

Lorie and Savage also consider the case where projects require capital outlays in several time periods, and they propose a method later shown by Weingartner [2] to suffer from several serious defects. Weingartner in [2] identifies capital rationing as an optimization problem and develops an integer programming model. This model, for the single period case, corresponds to the Dantzig formulation of the (0-1) *knapsack* problem, [3]. The model employed by Weingartner in the multiple outlay case corresponds to the (0-1) *multi-dimensional knapsack* problem (i.e., the knapsack problem with restrictions on weight, volume, height, etc.). Traditionally, the knapsack problem has been solved by dynamic programming and most recently by an enumerative technique developed by Gilmore and Gomory [4]. For the multidimensional knapsack problem, Weingartner and Ness [5] use a recursive relation to solve the complement problem (where projects are successively eliminated instead of accepted) and have reported interesting computational results. Shapiro and Wagner [6] have also studied these problems, demonstrating their connection with renewal problems formulated by means of recursive expressions.

Cord, [7] formulates the single period problem for the case of uncertain returns, and seeks to maximize the total return on investment while maintaining the average variance for the total investment within a certain predetermined value. Cord uses the method suggested by Bellman [8] of incorporating one constraint into the objective function by means of a Lagrange multiplier and then, with a single constraint left, applying the dynamic programming solution of the knapsack problem. A discussion of the drawbacks of the method, and the example problem of Cord, may be found in [9].

Finally, we point out that the present discounted value used by Lorie and Savage and by Weingartner has been a controversial issue due to the interest rate or "cost of capital" employed to obtain such discounted values. Baumol and Quandt [10] have indicated the serious difficulties that this approach entails, and have suggested an alternative objective function based on explicit discount rates and subjective utilities. Throughout this work, we shall assume subjective utility functions to express the corresponding figure of merit of the models to be derived.

3.2 THE VARIOUS CASES OF INVESTMENT DECISIONS

We shall consider various cases of investment decisions on independent projects confronting a firm or a government agency. We shall derive programming models in each case which may be interpreted as network flow problems on capacitated networks, with the additional constraint that flow on a subset of the arcs must be either zero or the upperbound on the arc. The solution techniques provided are special cases of the branch and bound algorithm presented in Chapter II. The problems to be analyzed and their characteristics are the following:

i) The capital investment problem requiring cash outlays in various time periods for each project is formulated as a maximum flow problem on a single-source single-sink capacitated network, where flow on the arcs represents cash flow and the flow on the arcs emanating from the source is restricted to be either zero or at upper bound. Its analogy to a special class of plant location problems is indicated. The branch and bound algorithm, as adapted to the problem, permits the use of the rounding operation; furthermore, the solution of the linear programming problem associated with each node of the solution tree may be obtained by simple inspection.

ii) The (0-1) knapsack problem is then considered as a special case of the previous problem. The solution proposed by Lorie and Savage (that of maximizing net present discounted value) is shown to represent the root node of the branch and bound tree.

Next part will be devoted to analysis of multistaged resource allocation problems where the horizon and staging are assumed to be given.

iii) The first of these problems to be considered is a capital budgeting problem requiring a single costs outlay per project and subject to capital rationing at each period; but the cash outlays, and thus the investment decision, may be deferred to a later period.

The resulting model which we shall call the *multi-knapsack* problem is studied, certain of its properties determined, and finally an equivalent network flow model on a bipartite graph is derived which resembles the fixed-charge transportation problem considered in Chapter VI. The branch and bound technique as applied to the problem, permits the use of the rounding operation; the linear program to be solved at each node of the solution tree is a capacitated transportation problem with surpluses and deficits and with certain routes prohibited.

iv) Finally, a special type of multi-knapsack problem is considered in which all items (projects) must be assigned to knapsacks of given capacity so as to minimize the number of knapsacks required to adequately allocate the items of the problem.

Problems iii) and iv), although presented within the framework of capital budgeting, arise in a variety of fields and in particular two such applications to optimal allocation of computer system facilities are discussed in detail

3.3 THE MULTIPERIOD CAPITAL INVESTMENT PROBLEM

Consider a government agency or a corporate division confronted with the problem of allocating a multi-staged budget with ceilings on each stage, among a set of independent projects requiring capital outlays in various time periods (Problem i) Government agencies typically face this problem when the available amount of capital is determined exogenously by legislature appropriation or by government budget planners. In the case of a corporate division, top management may determine the budgets and simultaneously cut off the division from acquiring additional funds from the capital market

Let B_j, $j=1,\ldots,n$ be the budget ceilings at each stage or time period and let $a_{ij} \geq 0$ be the capital outlay required by project i, $(i=1,\ldots,m)$ at time period j. Assume a certain utility f_i associated with the acceptance of project i*, the f_i might for example be subjectively determined by the decision maker. A set of projects must be selected for investment so that the total utility is maximized while maintaining the capital outlay at each stage within the corresponding budgetary ceiling.

We derive an analytical model by considering a bipartite network $G = [N_1, N_2, A]$, where N_1 is a set of m nodes each representing a project proposal, and N_2 a set of n nodes each associated with one of the stages considered. Let $\sum_{j=1}^{n} a_{ij}$ be the "demand" or input associated with node $i \in N_1$ and B_j the "demand" or output associated with node $j \in N_2$. Let x_{ij}, the flow on the arc $(i, j) \in A$, represent a capital outlay, and capacitate these arcs with the upper bounds a_{ij}. Then the capital allocation problem defined above may be expressed as follows: find a flow pattern on the network so as to

$$P' : \text{Maximize} \quad z = \sum_{i=1}^{n} f_i \, y_i \tag{3.1}$$

$$\text{Subject to} \quad \sum_{i=1}^{m} x_{ij} \leq B_j \qquad , j=1,\ldots,n \tag{3.2}$$

$$\sum_{j=1}^{n} x_{ij} = y_i \sum_{j=1}^{n} a_{ij}, \quad i=1,\ldots,m \tag{3.3}$$

$$0 \leq x_{ij} \leq a_{ij} \qquad , \forall (i,j) \in A \tag{3.4}$$

$$y_i \quad \text{integer} \tag{3.5}$$

where the y_i are decision variables associated with each node $i \in N_1$ which may take on the values 0 or 1 according to whether project i is

---

* f_i would represent the net present value of investing in project i, discounted by the appropriate rate of interest, if the total present value approach of Lorie and Savage is adopted.

rejected or accepted for investment. (By summing (3.4) over j and comparing the result with (3.3), we obtain $y_i \sum_{j=1}^{n} a_{ij} \leq \sum_{j=1}^{n} a_{ij}$; and since by (3.5) y_i is restricted to be integer, it follows that the only possible values for y_i are 0 or 1.)

Constraints (3.2) restrict the capital investments incident on node j to be within the available budget at period j. Constraints (3.3) indicate that if project i is accepted ($y_i = 1$) the sum of the flows leaving node i must be equal to the total investment required for that project over the entire horizon. Coupled with the upperbounding constraints (3.4), this condition forces the flow on arcs emanating from i to be at upper bound as expected. By the same reasoning, if $y_i = 0$, then $\sum_{j=1}^{n} x_{ij} = 0$ and the flows on the arcs x_{ij} are at zero level.

Observe that the capital budgeting problem as interpreted in this network flow context corresponds to a special class of plant location problems [11], [12]; however, in our problem we are maximizing, the flows from plants (projects) to destinations (time periods) are capacitated, and there is no explicit participation of the x_{ij} in the objective function.

Note that relations (3.3) permit P' to be exclusively expressed in terms of the set of variables x_{ij}, as follows:

$$P : \text{Maximize} \quad z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_i \, x_{ij} \tag{3.6}$$

$$\text{Subject to} \quad \sum_{i=1}^{m} x_{ij} \leq B_j \quad , \; j=1,\ldots,n \tag{3.7}$$

$$0 \leq x_{ij} \leq a_{ij} \; , \; \forall \, (i,j) \in A \tag{3.8}$$

$$y_i = \sum_{j=1}^{n} x_{ij} \Big/ \sum_{j=1}^{n} a_{ij} \quad \text{integer} \tag{3.9}$$

where $c_i = f_i \Big/ \sum_{j=1}^{n} a_{ij}$ represents the total utility of project i per unit of investment, and thus all arcs emanating from the same node i incur the same cost c_i

We observe in passing that if constraint (3.9) is deleted, the resulting problem may be decomposed into n mutually independent programs of the form Max $z_j = \sum_{i=1}^{n} c_i x_{ij}$, $\sum_{i=1}^{m} x_{ij} \le B_j$, $0 \le x_{ij} \le a_{ij}$, $\forall_i$, each one associated with time period j; the solution of which may be obtained by simple inspection as will be shown later.

Before proceeding to develop a solution technique for problem P, we shall show how the problem may be formulated as that of obtaining the maximum flow that maximizes total utility.

The bipartite network G with multiple sources and sinks may be transformed into an equivalent network with a single source and a single sink. This may be done by adding artificial nodes s and t, and artificial arcs (s,i), $\forall\ i \in N_1$ and $(j,t)\ \forall\ j \in N_2$, with the following associated values: $c_{si} = 0$, $u_{si} = \sum_{j=1}^{n} a_{ij}$ and $c_{jt} = 0$, $u_{jt} = B_j$; where u_{si} and u_{jt} denote the upperbounds on the respective arcs. Furthermore, we shall require that flow on arcs (s,i), $i \in N_1$ be either zero or otherwise that it saturates the arc. The associated network is shown in Fig. 3-1. The first number on each arc represents cost and the second represents arc capacity.

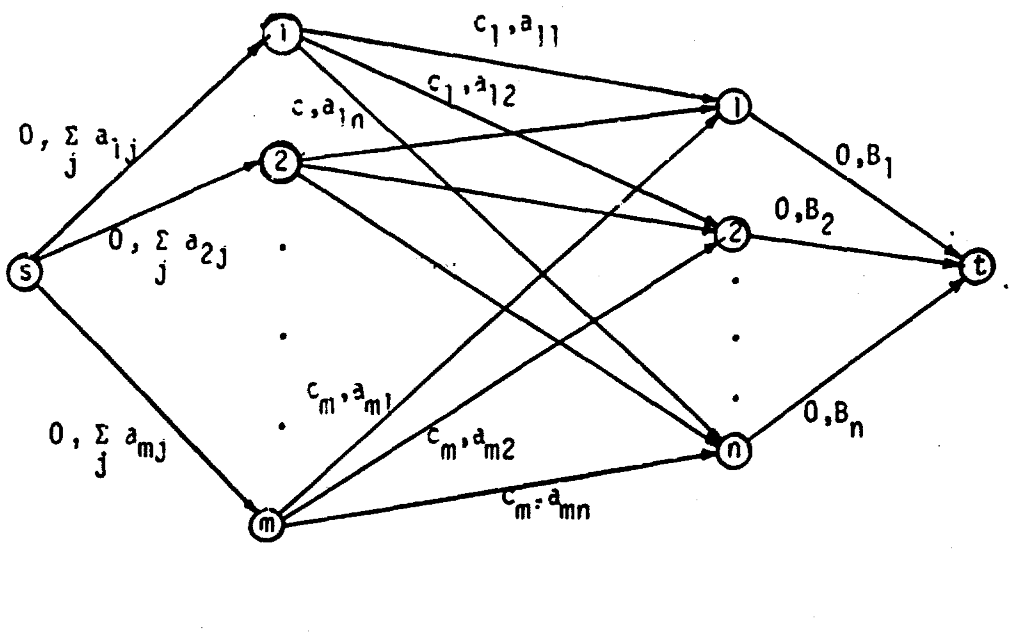


FIG. 3-1

The problem therefore may be expressed in terms of network flow theory as an *analysis* problem: find the maximum flow from s to t that maximizes cost on the network of Fig. 3-1, as well as its distribution pattern, such that arcs (s,i) are either not used or saturated.

The highly combinatorial nature of the problem does not permit network flow theory, in its current state of development, to provide a labeling technique (primal-dual method) to cope with such a problem. However, since a duality theory for discrete programming has recently been developed by Balas [13], a generalized concept of complementary slackness may be derived for this class of problems and thus a generalization of the out-of-kilter method [14] for networks with bivalent arcs may be developed. The author has been working on such an approach, but is unable at this point to present final successful results.

3.4 DEVELOPMENT OF A SOLUTION METHOD

We shall adapt in this section the branch and bound algorithm presented in Chapter II as applied to the solution of problem P. The notation to be employed complies with that used in Chapter II. We shall first define the sets S_1, T_1 and Ω_1 as follows:

$$S_1 = [x_{ij} \ / \ \sum_{i=1}^{m} x_{ij} \leq B_j, \ 0 \leq x_{ij} \leq a_{ij}] \tag{3.10}$$

$$T_1 = [x_{ij} \ / \ \sum_{j=1}^{n} x_{ij} + \sum_{j=1}^{n} a_{ij} = \text{integer}, \ x_{ij} = 0 \text{ or } a_{ij}, \ \forall (i,j)] \tag{3.11}$$

$$\Omega_1 = S_1 \cap T_1 \tag{3.12}$$

We observe that the sets thus defined satisfy the assumptions made in the original development. The set S_1 is a closed convex set in E^{m+n} obtained as the intersection of the hyperplanes (3.7) and (3.8); it is also bounded since each variable x_{ij}, from (3.8), is bounded above and below. T_1 is a non-empty set in the same space as S_1 (e.g. $x_{ij} = 0$, $\forall$ (i, j) ϵ T_1), and is also finite since $x_{ij} = 0$ or a_{ij}. Finally, from (3.12) Ω_1 is finite, since T_1 is finite.

Note also that since $a_{ij} \geq 0$ and $B_j \geq 0$, at least a feasible solution exists, namely, the *status quo*, i.e. the policy of zero investment; and thus an optimal solution always exists.

Branching Operation. Given a certain node ℓ of the solution tree with associated sets Ω_ℓ and S_ℓ, the branching is defined by their intersection with the sets

$$v_{\ell,r} = [x_{ij} / x_{kj} = 0, \forall_j] \Rightarrow y_k = 0 \qquad (3.13)$$

$$v_{\ell,r+1} = [x_{ij} / x_{kj} = a_{kj}, \forall_j] \Rightarrow y_k = 1 \qquad (3.14)$$

for a given $i = k$. The sets thus defined satisfy the sufficient conditions to form a partition of Ω_ℓ, (cf. theorem 2.1):

$$v_{\ell,r} \cap v_{\ell,r+1} = [x_{ij} / x_{kj} = 0, x_{kj} = a_{kj}, \forall_j] = \Phi \qquad (3.15)$$

$$v_{\ell,r} \cup v_{\ell,r+1} = [x_{ij} / x_{kj} = 0 \text{ or } a_{kj}, \forall_j] \qquad (3.16)$$

and since Ω_ℓ is a subset of Ω_1 (by branching operation) and from (3.10) to (3.12), the variables x_{kj} in Ω_ℓ may take on the values 0 or a_{kj}. Thus the intersection of Ω_ℓ with (3.16) is Ω_ℓ and the second condition for sufficiency is also satisfied.

Finally, since at each branching operation n variables x_{ij} are set either to zero or at upper bound, and since the number of variables is finite, eventually we will obtain a terminal node t with $S_t \equiv \Omega_t$ containing a single element of the domain Ω_1 and hence all feasible solutions to P may be enumerated in similar fashion by developing the entire solution tree.

3.5 THE AUXILIARY PROBLEM AND ITS SUBALGORITHM

At each iteration of the branch and bound algorithm, associated with each newly-generated node ℓ of the solution tree, a continuous *auxiliary* *problem* A_ℓ derived from P must be solved. Denote by $I_0 \subseteq N_1$ the subset of nodes of the network G for which $y_i = 0$, (i.e., nodes representing rejected projects); by $I_1 \subseteq N_1$ the subset of N_1 associated with $y_i = 1$, (accepted projects); and by $\bar{N}_1 = N_1 - I_0 - I_1$, the subset of projects that remain "free" to be accepted or rejected at this step of the solution process. Then the auxiliary problem A_ℓ takes the form

$$A_\ell : \text{Maximize} \quad z(\ell) = \sum_{i=1}^{m} \sum_{j=1}^{n} c_i x_{ij} \tag{3.17}$$

$$\text{Subject to} \quad \sum_{i=1}^{m} x_{ij} \le B_j \quad , j=1,\ldots,n \tag{3.18}$$

$$0 \le x_{ij} \le a_{ij} \quad , \forall (i,j) \tag{3.19}$$

$$y_i = \sum_{j=1}^{n} x_{ij} \Big/ \sum_{j=1}^{n} a_{ij} = 0 \quad , i \in I_0 \tag{3.20}$$

$$y_i = \sum_{j=1}^{n} x_{ij} \Big/ \sum_{j=1}^{n} a_{ij} = 1 \quad , i \in I_1 \tag{3.21}$$

This problem is a linear program whose solution may be obtained by inspection. Indeed, from (3.20), $x^*_{ij} = 0$, $i \in I_0$. From (3.21) and (3.19), $x^*_{ij} = a_{ij}$, $i \in I$. The problem (3.17), (3.18), (3.19) with the remaining free variables may be decomposed into n *mutually independent* programs of the form:

$$\text{Maximize} \quad z_j = \sum_{i \in \bar{N}_1} c_i x_{ij} \tag{3.22}$$

$$\text{Subject to} \quad \sum_{i \in \bar{N}_1} x_{ij} \le \bar{B}_j \tag{3.23}$$

$$0 \le x_{ij} \le a_{ij} \tag{3.24}$$

where $\bar{B}_j = B_j - \sum_{i \in I_1} a_{ij}$. We may assume without loss of generality that the indexing of projects $i \in N_1$ is done in decreasing order of their utility per unit of investment c_i, so that $c_1 \geq c_2 \geq \ldots \geq c_m$. Under this assumption it is obvious that the optimal solution to (3.22) - (3.20) may be obtained by simply setting the variables x_{ij} equal to their upper bound in the order of the index i until the budget $\bar{B}_j$ is exhausted. If $\sum_{i \in N_1} a_{ij} > \bar{B}_j$, then one single variable will take on a value less than its upperbound. The problem (3.22) - (3.24) may be solved for all j in this fashion. Thus the optimal solution to the auxiliary problem A_ℓ will be $x^*_{1j} = a_{1j}$ if $a_{1j} \leq \bar{B}_j$, zero otherwise and:

$$x^*_{rj} = \begin{cases} 0 & , \text{ if } r \in I_0 \\ a_{rj} & , \text{ if } r \in I_1 \\ a_{rj} & , \text{ if } \sum_{i=1}^{r-1} x^*_{ij} < \bar{B}_j \text{ and } \sum_{i=1}^{r} x^*_{ij} \leq \bar{B}_j,\ r \in \bar{N}_1 \ , r > 1 \\ \bar{B}_j - \sum_{i=1}^{r-1} x^*_{ij} & , \text{ if } \sum_{i=1}^{r-1} x^*_{ij} < \bar{B}_j \text{ and } \sum_{i=1}^{r} x^*_{ij} > \bar{B}_j,\ r \in \bar{N}_1 \ , r > 1 \\ 0 & , \text{ if } \sum_{i=1}^{r-1} x^*_{ij} \geq \bar{B}_j,\ r \in \bar{N}_1 \ , r > 1 \end{cases} \tag{3.25}$$

The objective function will have the following value:

$$z^*(\ell) = \sum_{i \in I_1} f_i + \sum_{i \in \bar{N}_1} \sum_j c_i x^*_{ij} \tag{3.26}$$

Note that if for any problem $\bar{B}_j < 0$, the corresponding A_ℓ is infeasible and the node of the tree may be excluded without further computation.

Rounding operation. Observe that from the optimal solution (3.25), (3.26) to A_ℓ, and from (3.9), the set of y_i^* may be determined. If all of them are integer, then (3.25), (3.26) constitute a feasible solution to problem P. If this is not the case, then a feasible solution to P may be obtained without any additional computational effort by simply setting

$$\hat{y}_i = \begin{cases} y_i^* \ , \text{ if } y_i^* = 0 \text{ or } 1 \\ 0 \ , \text{ if } 0 < y_i^* < 1 \end{cases} \tag{3.27}$$

or equivalently, if $0 < x_{ij}^* < a_{ij}$, then set $\hat{x}_{ij} = 0$ for all $j \in N_2$, otherwise set $\hat{x}_{ij} = x_{ij}^*$. The value of the objective function is given by:

$$\hat{z}(\ell) = \sum_{i \in I_1} f_i + \sum_{i \in \bar{N}_1} \sum_j c_i \hat{x}_{ij} \tag{3.28}$$

The solution thus obtained is feasible for P; note that in obtaining $\hat{x}_{ij}$, the values x_{ij}^* have been reduced if changed at all, hence constraints (3.7) and (3.8) are still satisfied. Also from (3.27), constraint (3.9) is satisfied and the solution is feasible for P.

The rounding operation defined by (3.27) and (3.28) will therefore permit us to perform rejection of certain branches of the branch and bound tree, since (3.28) constitutes a lower bound on the optimal solution to P.

3.6 THE BRANCH AND BOUND ALGORITHM

Having shown that the assumptions of the branch and bound algorithm are satisfied and having developed a subalgorithm for solution of the auxiliary problem, we may now proceed to establish the solution method as applied to our capital budgeting problem. Note that we have simplified the statement of the algorithm (cf. Chapter II) and also have expressed it in terms of a maximization problem.

STEP 1. Set $i = 1$, generate node 1 by solving A_1 (i.e., P without constraints (3.9)), and let z^*, x^*_{ij} be the optimal solution. From (3.9) obtain y^*_i. If all y^*_i are 0 or 1, stop; the solution is optimal. Otherwise bound node 1 with $U_1 = z^*$. *Round* node 1 to obtain $(\hat{z}, \hat{x}_{ij})$. Set $L_1 = \hat{z}$. If $L_1 = U_1$, stop; the rounded solution is optimal. Otherwise $L_1 < U_1$. Set $i = i + 1$ and go to step i.

STEP i a) BRANCH. Branch from bounded node ℓ. Select one y^*_k having a fractional value. Create nodes r and r + 1 and directed arcs (ℓ, r) and $(\ell, r+1)$. Solve problem A_r with $y_k = 0$, adding k to set I_0, and solve problem A_{r+1} with $y_k = 1$, adding k to set I_1. If $z^*(r)$ or $z^*(r+1) < L_{i-1}$, *reject* the corresponding node. If one is infeasible, *exclude* the corresponding node.

b) ROUND. Round nodes r and r + 1.

c.1) BOUND FROM BELOW. Set $L_i = \max\,[L_{i-1}, \hat{z}(r), \hat{z}(r+1)]$. Reject all nodes with $z^* < L_i$.

c.2) BOUND FROM ABOVE. Select node ℓ such that $z^*(\ell) = \max\,\{z^*(k)\}$, for current terminal nodes. Upperbound node ℓ with $U^k_i = z^*(\ell)$. If $L_i = U_i$, stop; the feasible solution that provides the lower bound is optimal. Otherwise, $L_i < U_i$. Set $i = i + 1$ and go to step i.

At each branching operation, one of the current functional y^*_i must be selected to take on the values 0 and 1. There may be several such variables and in general there is no clear cut selection rule that would guarantee the fastest convergence to the optimum. Usually certain heuristic rules are discovered when sufficient computational experience with the algorithm is available. Our experience with problems solved by hand has indicated that selection of the fractional y^*_r having the largest total investment $\sum_j a_{rj}$ tends to result in infeasible nodes for the branch $y_r = 1$, thus reducing the number of terminal nodes for which data must be preserved and resulting in a reduced time of computation.

3.7 SOLUTION OF AN EXAMPLE PROBLEM

Consider as an example the problem given in Table 3-1, taken from [2], involving 10 projects and 2 stages. The budgetary ceilings are $B_1 = 50$, $B_2 = 20$.

Project No	f_i	a_{i1}	a_{i2}	$\frac{f_i}{\sum_j a_{ij}}$	Project No	f_i	a_{i1}	a_{i2}	$\frac{f_i}{\sum_j a_{ij}}$
1	15	6	2	1.875	6	40	30	35	0.615
2	17	6	6	1.417	7	12	18	3	0.571
3	15	6	7	1.154	8	17	54	7	0.279
4	12	6	6	1.000	9	14	48	4	0.269
5	14	12	3	0.933	10	10	36	3	0.256

TABLE 3-1

To generate the root, (node 1), the auxiliary problem A_1 must be solved. The solution is obtained for each time period by means of expressions (3.25). Here $I_0 = I_1 = \Phi$ and $\bar{B}_j = B_j$. The solutions are:

$$x^*_{i1} = [6, 6, 6, 6, 12, 14, 0, 0, 0, 0]$$

$$x^*_{i2} = [2, 6, 7, 5, 0, 0, 0, 0, 0, 0], \text{ thus}$$

$$y^*_1 = [1, 1, 1, \frac{11}{12}, \frac{12}{15}, \frac{14}{65}, 0, 0, 0, 0], \; z^*(1) = 47 + 30.82 = 77.82$$

and by rounding y^*_1 we obtain

$$\hat{y}_i = [1, 1, 1, 0, 0, 0, 0, 0, 0, 0], \; \hat{z}(1) = 47$$

Therefore, $L_1 = 47$, $U_1 = 77.82$ and node 1 is bounded.

The subsequent iterations and their pertinent data are shown

Step i	Feasible nodes $F(i)$	Excluded nodes	Rejected nodes	Lower bound L_i	Upper bound U_i
1	1^*			47	77.82†
2	3^*	2		47	77.19
3	4^*, 5			59	74.46
4	5, 6^*, 7			59	73.85
5	5, 8^*, 9		7	70	73.59
6	5, 9, 10^*		11	70	73.13
7	5^*, 9, 13	12		70	73.10†
8	9, 13, 15^*	14		70	72.96
9	9, 13, 17^*		16	70	72.78
10	9, 13^*		18, 19	70	71.68†
11	9^*	20	21	70	70.56†
12	23^*	22		70	70.54
13	25^*	24		70	70.51
14	27^*	26		70	70

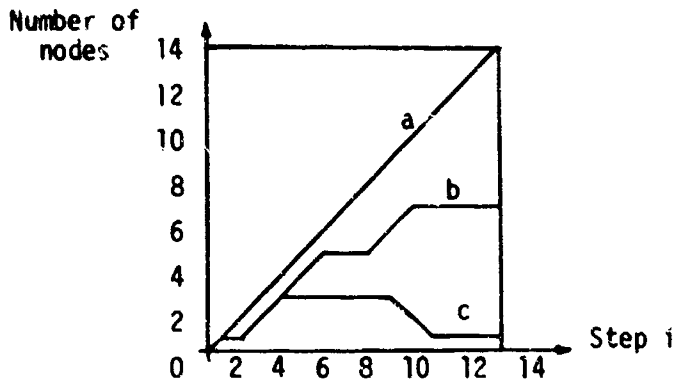
* : bounded node

† : search on a new branch of the solution tree

TABLE 3-2

in Table 3-2 and the actual solution tree in Figure 3-3 A total of 27 nodes are generated although only 20 need be evaluated by means of expressions (3 27), (The infeasibility of excluded nodes is detected when $\bar{B}_j < 0$ for any j) The second column of Table 3-2 indicates the number of current terminal nodes for which information must be stored for later use. Note that at any one time no more than three such feasible terminal nodes exist

Fig. 3-2 graphically shows the effectiveness of the rejection rule. The difference between the lines a and b may be attributed to the selection rule employed in choosing the fractional variable y_i to be fixed at each iteration. The difference between lines b and c represents the effect of the rejection rule, which in this case not only dampens the growth rate of the set of feasible nodes, but in a certain interval makes the size of the set decrease with the number of iterations. Finally, note that since the number of feasible terminal nodes at the end of iteration 14 is only one; this indicates that the optimal solution, accept projects 1, 2, 4, 5 and 7, is unique.



a : terminal nodes of the solution tree

b : terminal nodes excluding infeasible nodes

c . terminal nodes excluding infeasible and rejected nodes

FIG. 3-2

In table 3-2 we have indicated the steps at which a search along a new branch of the tree is started. This type of information is valuable in the context of computer implementation of the algorithm. In fact, at each iteration, a search over all currently feasible nodes must be

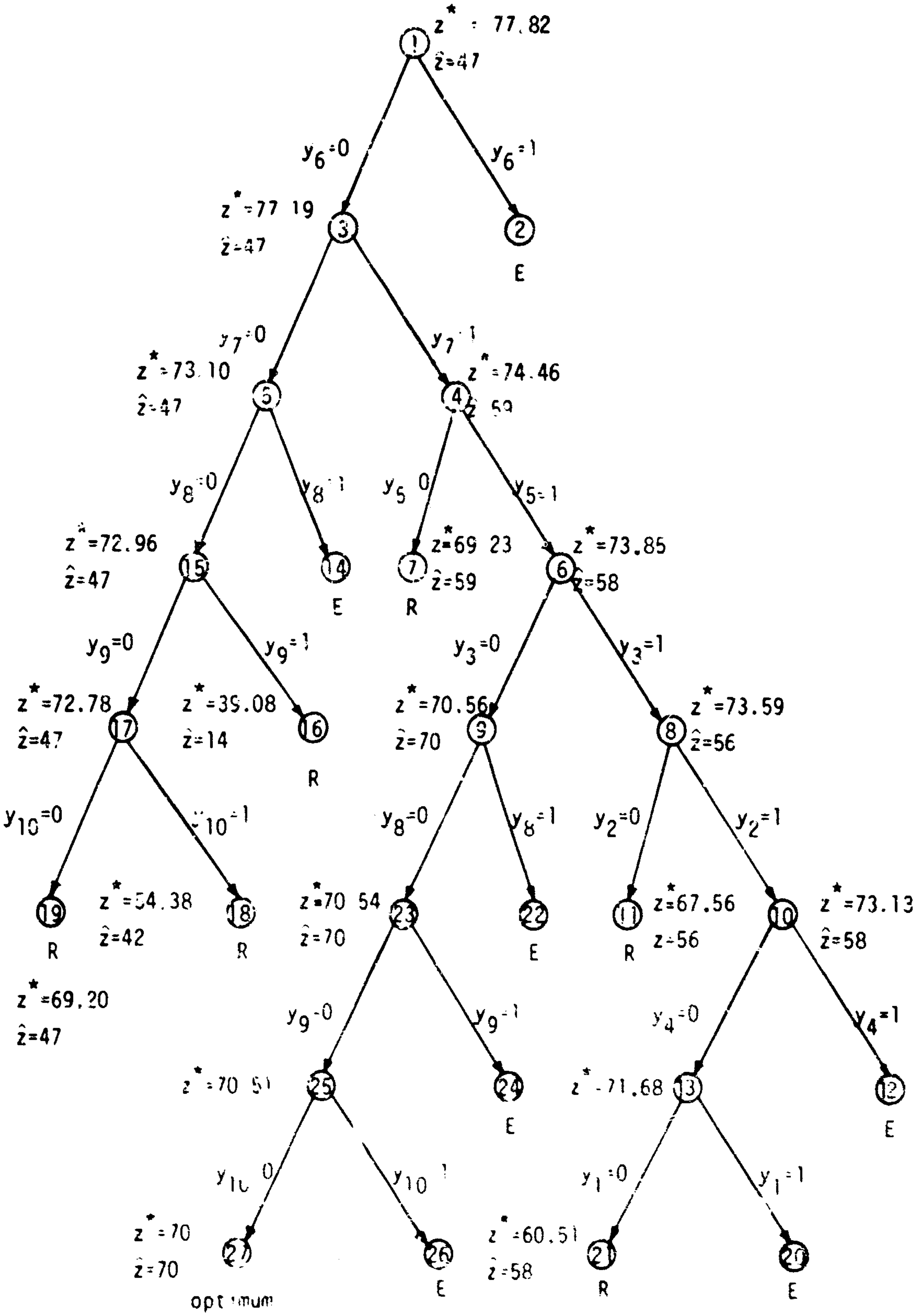


FIG 3-3 The Solution Tree

performed to determine the node from which to branch next: $\max_{k \in F(i)} \{z^*(k)\}$. This is essentially an optimization problem, solved by a table look-up equivalent to an exhaustive search. If at each iteration the node numbers of the best and second best (or as many as desired) z^* values are stored separately, and updated at each iteration, the number of table look-ups is reduced. It is easy to verify that a table look-up is not required until after the number of tree branches so far developed at least equals the number of decreasingly best nodes stored separately. At this point, the table look-up would produce the next set of decreasingly best nodes needed to begin the next cycle of the operation.

In the example considered, if the three values with the best z^* values are available, the first table look-up would be required at iteration 12 to determine that node 23 should be the one from which to branch next. At that point, of course, the table consists of one single element.

It may be worthwhile to point out that the dimensionality of the tree depends largely on the number of projects considered rather than on the number of time periods; the latter only implies extra computation of expressions (3.25) for all time periods at each node of the tree. That is, the total number of nodes of the final tree is expected to vary slightly as a function of the number of time periods considered.

Excellent results have been obtained with this branch and bound technique. For a report on this computational experience, the reader is referred to [15].

3.8 NOTES TO CHAPTER III

[1] Lorie, J. H., and L. J. Savage, "Three Problems in Rationing Capital", *The Journal of Business*, Vol. XXVIII, October 1955, pp. 229-239.

[2] Weingartner, H. M., *Mathematical Programming and the Analysis of Capital Budgeting Problems*, Prentice-Hall, Inc., Englewood Cliffs, 1963.

[3] Dantzig, G. B., "Discrete-Variable Extremum Problems", *Operations Research*, April 1957, pp. 266-277.

[4] Gilmore, P. C. and R. E. Gomory, "A Linear Programming Approach to the Cutting Stock Problem - Part II", *Operations Research*, Sept. - Oct., 1963, pp. 863-888.

[5] Weingartner, H. M. and D. N. Ness, "Methods for the Solution of the Multi-dimensional 0/1 Knapsack Problem", *Operations Research*, Jan. - Feb., 1967, pp. 83-103.

[6] Shapiro, J. F. and H. M. Wagner, "A Finite Renewal Algorithm for the Knapsack and Turnpike Models", *Operations Research*, March-April, 1967, pp. 319-341.

[7] Cord, J., "A Method for Allocating Funds to Investment Projects when Returns are Subject to Uncertainty", *Management Science*, January, 1964, pp. 335-341.

[8] Bellman, R., "Comment on Dantzig's Paper on Discrete-Variable Extremum Problems", *Operations Research*, Oct. 1957, pp. 723-724.

[9] Weingartner, H. M., "Capital Budgeting of Interrelated Projects: Survey and Synthesis", *Management Science*, March 1966, pp. 485-516.

[10] Baumol, W. J. and R. E. Quandt, "Mathematical Programming and the Discount Rate under Capital Rationing", *Economic Journal*, June, 1965, pp. 317-329.

[11] Balinski, M. L., "Integer Programming: Methods, Uses, Computation", *Management Science*, November 1965, pp. 286-290.

[12] Efroymson, M. A. and T. L. Ray, "A Branch and Bound Algorithm for Plant Location", *Operations Research*, May-June, 1966, pp. 361-368.

[13] Balas, E., "Duality in Discrete Programming", Technical Report 67-5, *Operations Research House*, Stanford University, August 1967.

[14] Fulkerson, D. R., "An Out-of-Kilter Method for Minimal Cost Flow Problems", *Journal of the Society for Industrial and Applied Mathematics*, V. 9, 1961, pp. 18-27.

[15] Ochoa-Rosso, F. and K. G. Follansbee, A *Branch and Bound Algorithm for the Multiperiod Capital Investment Problem*, Research Report R68-40, Massachusetts Institute of Technology, Department of Civil Engineering, July 1968.

CHAPTER IV

SINGLE STAGE INVESTMENT: THE KNAPSACK PROBLEM

4.1 THE (0 1) KNAPSACK PROBLEM

The multi-dimensional knapsack problem studied in Chapter III, for the special cases $n = 1$ (one single period) corresponds to the (0-1) knapsack problem formulated by Dantzig. As we mentioned before, this problem has traditionally been solved by dynamic programming. Gilmore and Gomory [1] have recently developed a theory for *knapsack functions* (i.e., the optimum solution $z^0(B)$ is a knapsack function of the budget, which is considered as a parameter). Although they have presented algorithms for the knapsack problems without upperbound on the variables, for the (0-1) case they have only indicated how an algorithm based on dynamic programming should be derived to solve the problem for various values of the budget.

In this section and forthcoming sections we shall study some important economic interpretations related to the dual of the knapsack problem. In this context, we shall first relax the discrete restriction on the variables, and later we shall also formulate the dual of the original knapsack problem, making use of Balas' discrete programming duality theory. Two branch and bound solution methods will be proposed based on the general algorithm of Chapter II. The first method constitutes a special case of the algorithm developed for the multi-dimensional knapsack problem in Section 3.6. We shall discuss the simplifications resulting from assuming an investment horizon of one time period. The second algorithm, although a branch and bound solution method complying with the general theory of Chapter II, is closely related to the *additive algorithm* of Balas [2]. In addition, this algorithm is *parametric*, in the sense indicated by Ichbiah [3], (i.e., the tree simultaneously solves the knapsack problem for various values of the budget).

4.2 ECONOMIC INTERPRETATION OF THE DUAL LINEAR PROGRAM

By collapsing the index j in the formulation P of Section 3.3, the knapsack problem of Dantzig may be modeled by the following discrete (bivalent) programming problem:

$$Q : \text{Maximize} \quad z = \sum_{i=1}^{m} c_i \, x_i \tag{4.1}$$

$$\text{Subject to} \quad \sum_{i=1}^{m} x_i \le B \tag{4.2}$$

$$0 \le x_i \le a_i \quad , \forall_i \tag{4.3}$$

$$x_i = 0 \text{ or } a_i \quad , \forall_i \tag{4.4}$$

where (4.3) is obviously redundant, but has not been removed here for reasons that will become evident. Let us assume that constraint (4.4) is relaxed; that is, projects may be accepted for which the capital outlay is only a fraction of the total capital required, a_i. Under this assumption the resulting problem, which we shall denote Q', is a linear program whose associated dual program is the following problem D':

$$D' : \text{Minimize} \quad z' = B\,v + \sum_{i=1}^{m} a_i \, u_i \tag{4.5}$$

$$\text{Subject to} \quad v + u_i \ge c_i \tag{4.6}$$

$$v, u_i \ge 0 \quad , \forall_i \tag{4.7}$$

where v is the dual variable or *shadow price* associated with constraint (4.2) and the u_i the shadow prices associated with constraints (4.4). Let us assume that $c_i \equiv f_i \,/\, a_i > 0$, and that as before, the c_i are ordered in decreasing sequence, (i.e., $c_1 \ge c_2 \ge \dots \ge c_m$). Under these assumptions it is obvious that the optimal solution of Q' may be determined recursively by

$$x_1^* = \begin{cases} a_1 & , \text{ if } a_1 \leq B , \\ B & , \text{ if } a_1 > B , \end{cases}$$

$$x_r^* = \begin{cases} a_r & , \text{ if } \sum_{i=1}^{r-1} x_i^* < B \text{ and } \sum_{i=1}^{r} x_i^* \leq B, \ r > 1 \\ B - \sum_{i=1}^{r-1} x_i^* & , \text{ if } \sum_{i=1}^{r-1} x_i^* < B \text{ and } \sum_{i=1}^{r} x_i^* > B, \ r > 1 \\ 0 & , \text{ if } \sum_{i=1}^{r-1} x_i^* \geq B, \ r > 1 \end{cases} \tag{4.8}$$

Denoting by $r = s$ the index of the last accepted project, we see from (4.8) that only x_s may be less than a_s, and thus project s is the only one partially accepted. The rest are totally accepted if $r < s$ or totally rejected if $r > s$. From (4.8), the optimal solution will satisfy (4.2) as a strict equality (the constraint is active).

If $\underline{x}^*$ and v^*, $\underline{u}^*$ are optimal solutions in their respective programs, from duality theory of linear programming the following *complementary slackness* conditions must be satisfied:

$$v^* [B - \sum_{i=1}^{m} x_i^*] = 0 \tag{4.9}$$

$$u_i^* [a_i - x_i^*] = 0 \tag{4.10}$$

$$x_i^* [u_i^* + v^* - c_i] = 0 \tag{4.11}$$

From (4.9), if $v^* > 0$ then $\sum_{i=1}^{m} x_i^* = B$, which is the case under consideration. Then we conclude that for an optimal solution, the budget ceiling is a scarce resource and v^* may be interpreted as the value or imputed rate of an additional dollar added to the budget. Note that this rate is always positive* in the linear programming case, and its value will be determined below.

From (4.10), $u_i^* = 0$ for rejected projects, and for positive u_i^* the project is accepted. The values u_i^* represent the internal rate of return of one dollar invested in project i. From (4.11), for accepted projects the following holds:

$$u_i^* = c_i - v^* \geq 0 \ , \ i \leq s \tag{4.12}$$

We shall now proceed to determine the optimal solution v^*, $\underline{u}^*$ to the dual problem. For an optimal solution, both objective functions are equal, and taking into account the fact that $x_i^* = 0$ and $u_i^* = 0$ for $i > s$, (i.e., for rejected projects) we obtain:

$$\sum_{i=1}^{s-1} c_i \ a_i + c_s \ (B - \sum_{i=1}^{s-1} a_i) = B \ v^* + \sum_{i=1}^{s-1} a_i \ u_i^* \tag{4.13}$$

By substituting (4.12) in (4.13) and solving for v^* the following condition results:

$$v^* = c_s \tag{4.14}$$

and thus the optimal value of an additional dollar added to the budget is equal to the net present value per dollar invested of the last project accepted for investment in the optimal solution to Q. Finally, substituting (4.14) back in (4.12),

$$u_i^* = c_i - c_s \ , \quad i \leq s \tag{4.15}$$

which will be non negative since we have assumed $c_i \geq c_s$ for $i \leq s$. The rate of return of one dollar invested in project, i, $i \leq s$, is the

*Obviously $v^* = 0$ for the trivial case $\sum_{i=1}^{m} a_i < B$.

difference, if any, between the net present values per dollar invested of project i and project s (the marginally accepted project).

The values u_i^* suggest a natural way to define investment priorities. Lorie and Savage [4] propose a ranking strategy based on decreasing net present values per dollar of outlay, that is in decreasing order of their c_i. Thus, the ranking suggested by (4.15) and the one of Lorie and Savage would result in the same projects selected for investment. We emphasize again that this holds only for situations in which the marginal project accepted may be accepted as a fraction. We shall consider now the dual of problem Q, for which attempting project divisibility is an absolute *faux pas*.

4.3 THE DUAL OF THE DISCRETE PROGRAM

In this section we shall study the dual of the all-bivalent program Q based on the duality theory of discrete programming, [5]. We shall reconcile this theory with the dual program suggested by Weingartner [6] for the single period investment problem.

Let us consider Q subject to constraints (4.2) and (4.4) only, and drop the redundant constraints (4.3) from further consideration. The dual of such program is the following max-min problem:

$$D : \quad \underset{\underline{x}}{\text{Max}} \; \underset{v}{\text{Min}} \quad z' = B\,v - \sum_{i=1}^{m} s_i\, x_i$$

$$\text{Subject to} \quad v - s_i = c_i \tag{4.16}$$

$$v \geq 0 \;, \; x_i = 0 \text{ or } a_i \;, \; \forall_i$$

$$s_i \;\; \text{unrestricted}$$

Since the slack variables s_i are unrestricted, they effectively nullify constraints (4.16) and problem D may be rewritten as

$$D : \quad \underset{v}{\text{Min}} \left[B\,v - \underset{\underline{x} \in S_0}{\text{Max}} \left\{ \sum_{i=1}^{m} (v - c_i)\, x_i \;/\; v > 0 \right\} \right] \tag{4.17}$$

where $s_0 = \{\underline{x} / x_i = 0 \text{ or } a_i\}$. As suggested by Balas, the solution of problem Q may be obtained by solving the equivalent problem (4.17) using the partitioning technique of Benders (cf. Appendix A), thus obtaining the optimum $\underline{x}^*$ and v^*. However, we shall assume here that an optimal solution $\underline{x}^*$ to problem Q is already available (obtained for example by the branch and bound technique of Section 3.6).

The saddle-point theorem of Balas guarantees that if an optimal solution $\underline{x}^*$, z^*, to Q exists, then an optimal solution v^*, $\underline{s}^*$, z'^* to D exists, with $z^* = z'^*$.

THEOREM 4.1 (*Complementary Slackness*). *If* $\underline{x}^*$, z^* *and* v^*, z'^* *are optimal solutions to* Q *and* D *respectively, then*

$$v^* [B - \sum_{i=1}^{m} x_i^*] = 0 \tag{4.18}$$

Proof: Since by the saddle point theorem $z^* = z'^*$, we have

$$\sum_{i=1}^{m} c_i x_i^* = B v^* - \sum_{i=1}^{m} (v^* - c_i) x_i^*$$

$$\text{or} \quad 0 = v^* [B - \sum_{i=1}^{m} x_i^*]$$

So for $v^* > 0$, the budget is a scarce resource. Alternatively, if the budget is a free good, $\sum_{i=1}^{m} x_i^* < B$ and therefore $v^* = 0$. The dual variable v^* may be interpreted as in the continuous case: it is the value or imputed rate of an additional dollar added to the budget.

Once an optimal solution $\underline{x}^*$, z^* to Q is available, then the optimal dual variables may be determined as follows: If $\sum_{i=1}^{m} x_i^* < B$, then from theorem 4.1 it follows that $v^* = 0$ and thus $s_i^* = - c_i$.

If on the other hand, $\sum_{i=1}^{m} x_i^* = B$, then by substituting x_i^* for x_i in D we obtain the following square system of equations of order m + 1:

$$B\,v^* - \sum_{i=1}^{m} s_i^*\, x_i^* = z^* \tag{4.19}$$

$$v^* - s_i^* = c_i \qquad , \forall_i \tag{4.20}$$

$$v^* \geq 0 \tag{4.21}$$

LEMMA 4.1 *For an optimal solution x_i^* to Q, satisfying $\sum_{i=1}^{m} x_i^* = B$, the equation (4.19) is redundant.*

Proof· Multiplying each member of (4.20) by x_i^* and summing over i we have

$$v^* \sum_{i=1}^{m} x_i^* - \sum_{i=1}^{m} s_i^*\, x_i^* = \sum_{i=1}^{m} c_i\, x_i^*$$

but this expression, under the assumption that $\sum_{i=1}^{m} x_i^* = B$, is equal to (4.19).

It follows from the lemma that the rank of the system (4.20)-(4.21) is m (the system is triangular), and thus v^* may take any non-negative arbitrary value.

We shall select for v^* the value c_s where s is again the project among those accepted which has the minimum net present value per dollar invested. This choice will insure that all accepted projects have non-negative benefit, that is, $c_i - v^* \geq 0$. Thus, denoting by I_0 the set of rejected projects and by I_a the accepted set, we have

$$\begin{aligned} -s_i^* &= c_i - c_s \geq 0 \;, \; i \in I_a \\ -s_i^* &= c_i - c_s \gtrless 0 \;, \; i \in I_0 \end{aligned} \tag{4.22}$$

Observe that it may still be possible that for a rejected project $c_i > c_s$ implying that this project has a positive benefit ($c_i - c_s > 0$).

We may contrast (4.22) with the linear programming case studied in Section 4.2 where the following conditions were satisfied:

$$\begin{aligned} c_i - c_s &\geq 0 , \quad i \in I_a \\ c_i - c_s &\leq 0 , \quad i \in I_0 \end{aligned} \tag{4.23}$$

Thus in the integer programming context, as Weingartner remarks, rejected projects for which $c_i - c_s > 0$ are essentially taxed or penalized to eliminate any profit on them, thereby preventing their acceptance.

Finally we shall reconcile the results of the integer duality theory developed above with Weingartner's "alternate dual values" approach to establishing shadow prices in the integer capital investment problem. He assumes ([6], pp. 103-106) that an optimal solution $\underline{x}^*$, z^* to the primal integer program Q is available (i.e., the sets I_0 and I_a are given). He evaluates the "alternate dual variables" by solving a linear programming model constructed in such a way that it allows negative benefits only for rejected projects. The model he proposes, re-expressed in our terminology and notation, is as follows:

$$w : \text{Minimize} \quad Z = Bv + \sum_{i\in I_a} \upsilon_i x_i^*$$

$$\text{Subject to} \quad v + \upsilon_i \geq c_i \quad , \quad i \in I_a \tag{4.24}$$

$$v + \upsilon_i - \gamma_i \geq c_i , \quad i \in I_0 \tag{4.25}$$

$$v, \upsilon_i, \gamma_i \geq 0$$

where the benefit ($c_i - v$) is given by $\upsilon_i \geq 0$ for accepted projects and by ($\upsilon_i - \gamma_i$) for rejected projects. Under the assumption that $a_i \geq 0$ and

$c_i \geq 0$ and that $I_a \neq \Phi$, Weingartner's problem may be solved by inspection as warranted by the following theorem.

THEOREM 4.2 *The values v^*, υ_i^*, γ_i^* constitute an optimal solution to w only if $v^* + \upsilon_i^* = c_i$, $i \in I_a$ and $v^* + \upsilon_i^* - \gamma_i^* = c_i$, $i \in I_0$.*

Proof: For any $v^* > 0$, υ_i^* and γ_i^*, $i \in I_0$ exist satisfying (4.25) as a strict equality without altering the value of the objective function. Now consider constraints (4.24):

a) $v^* = 0$, $\upsilon_i^* = 0$, $\forall_i \in I_a$, is not possible since I_a is assumed non-empty.

b) if $v^* = 0$, then $\upsilon_i^* > 0$. (If $\upsilon_i^* = 0$ for a subset of I_a, then $c_i = 0$, or else the solution v^*, υ_i^* is infeasible.) Assume also $v^* + \upsilon_i^* > c_i$, $i \in I_a$. Then each υ_i^* may be decreased without violating the constraints (4.24) but with a decrease in the value of the objective function. Thus $v^* + \upsilon_i^* = c_i$.

c) $v^* > 0$, $\upsilon_i^* = 0$ for some $i \in I_a$ and $\upsilon_i^* > 0$ for the remaining $i \in I_a$. Assume also $v^* > c_i$ for all i such that $\upsilon_i^* = 0$. Then v^* may be decreased without violating the constraints, causing the objective function to decrease. Therefore $v^* = c_i$. If $v^* + \upsilon_i^* > 0$ still holds for i such that $\upsilon_i^* > 0$, then υ_i^* may be decreased without violating the constraint, causing the objective to decrease. Thus $v^* + \upsilon_i^* = c_i$. Q.E.D.

Furthermore, by taking the dual of w it can be shown that the optimal solution $\hat{z}^*$ to w is equal to the assumed known value z^* of the integer problem Q. Thus the solution of w is reduced to solving the system of equations:

$$Bv^* + \sum_{i \in I_a} \upsilon_i^* x_i^* = z^* \tag{4.26}$$

$$v^* + \upsilon_i^* = c_i \quad , \quad i \in I_a$$

$$v^* + \upsilon_i^* - \gamma_i^* = c_i \quad , \quad i \in I_0$$

$$v^*, \upsilon_i^*, \gamma_i^* \geq 0$$

This system is solved as follows: If $\sum_{i \in I_a} x_i^* \quad B$, then (4.26) is redundant*; then by selecting $v^* = c_s$ arbitrarily with c_s being the smallest c_i, $i - I$, the solution is

$$v^* = c_s$$

$$\upsilon_i^* = c_i - c_s \geq 0 \qquad i \in I_a \tag{4.27}$$

$$\upsilon_i^* - \gamma_i^* = c_i - c_s \gtrless 0 \quad , \quad i \in I_0$$

Comparing now the results of (4.22), obtained by direct exploitation of discrete duality theory, with (4.27), derived from Weingertner's intuitively constructed model w, we observe that they are the same, thus establishing the equivalence of both approaches in determining a system of shadow prices for the capital investment problem under consideration.

---

* If $\sum_{i \in I_a} x_i^* < B$, solution is $v^* = 0$, $\upsilon_i^* = c_i$, $i \in I_a$, $\upsilon_i^* - \gamma_i^* = c_i$, $i \in I_0$.

4.4 SOLUTION OF THE KNAPSACK PROBLEM

The branch and bound algorithm of Section 3.6 may be directly applied to solve the knapsack problem as formulated in problem Q. However, its application to this single period investment problem is totally deterministic in the sense that when a branching operation is about to take place from a bounded node, the solution of the auxiliary problem for that node contains only <u>one</u> varialbe with $0 < x_i^* < a_i$. Hence this particular variable must be *a fortiori* be fixed to its only possible values $x_i = 0$ and $x_i = a_i$ in order to continue the execution of the algorithm.

To illustrate its application and to compare it with the alternative algorithm of Section 4.5, consider the following problem.

EXAMPLE

Assume that the ten projects of the example in Section 3.7 are considered for investment with the same payoffs and with the same total outlays required except that these outlays must be invested in a single time period. Table 4-1 shows the pertinent data. The projects are again ordered in decreasing values of their c_i. The budgetary ceiling is B = 70.

To initialize the problem and thus generate the root-node 1, problem Q is solved ignoring constraints (4.4). We observe in passing that the solution proposed by Lorie and Savage corresponds to the solution of the auxiliary problem associated with the root of the solution tree. This solution is $z^* = 79.15$ and $x^* = [8, 12, 13, 12, 15, 10, 0, 0, 0, 0]$, where $x_6^* = 10$ is the only variable root zero or at upper bound; thus

Project No	f_i	a_i	$\frac{f_i}{a_i}$	Project No	f_i	a_i	$\frac{f_i}{a_i}$
1	15	8	1.875	6	40	65	0.615
2	17	12	1.417	7	12	21	0.571
3	15	13	1.154	8	17	61	0.279
4	12	12	1.000	9	14	52	0.269
5	14	15	0.933	10	10	39	0.256

TABLE 4.1

the branching will take place from node 1 by fixing $x_6 = 0$ and $x_6 = 65$. The rounding solution associated with the node is $\hat{x} = [8, 12, 13, 12, 15, 0, 0, 0, 0, 0]$ with $\hat{z} = 73$. The optimal solutions of nodes 11 and 21 are obtained after 11 iterations of the algorithm. The pertinent information at each iteration is recorded in Table 4-2 and the solution tree in Fig. 4-2. We observe that since the rounding operation at node 1 produces a solution which turns out to be optimal, the trimming of the solution tree by use of the rejection operation is very powerful for this particular example.

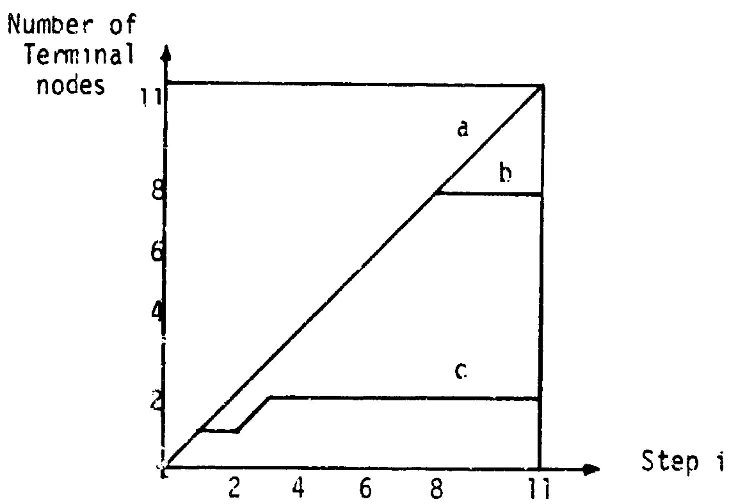
STEP i	Feasible nodes $F(i)$	Excluded nodes	Rejected nodes $R(i)$	Lower bound L_i	Upper bound U_i
1	1*			73	79.15†
2	3*		2	73	78.71
3	4 , 5*			73	75.79
4	4 , 7*		6	73	75.69
5	4 , 9*		8	73	75.56
6	4*, 11		10	73	74.73†
7	11 , 12*		13	73	74
8	11 , 15*		14	73	73.28
9	11 , 17*	16		73	73.27
10	11 , 19*	18		73	73.26
11	11 , 21	20		73	73

* : bounded node

† : search on a new branch of the solution tree

TABLE 4-2

From Table 4-2 we observe that information for no more than two nodes need be stored at any step of the solution process. The number of nodes remaining in the feasible set at the last iteration, F(11), indicate the number of optimal solutions. For this example two strategies, both not exhausting the budget, are available: a) accept projects 1, 2, 3, 4 and 5; b) accept projects 1, 2, 3, 5, 7. Note that the optimal solution has a total utility of 73, as opposed to 70 in the example of Section 3.7 where the budget ceilings were given in two time periods.



a : C(i), b : F(i)$\cup$R(i), c : F(i)

FIG. 4-1

In Fig. 4-1, the difference between the lines (a) and (b) is fixed since the branching operation is deterministic. Line (c) indicates the number of terminal nodes to be stored at each iteration. The difference between lines (b) and (c) is the result of the rejection operation of the algorithm. Note that the rate of increase of terminal nodes with number of iterations performed is drastically reduced by the rejection rule in this case.

According to the theory of Section 4.3, the imputed dual prices would be $v^* = 0$, since the budget is a free good and $s_i^* = -c_i$.

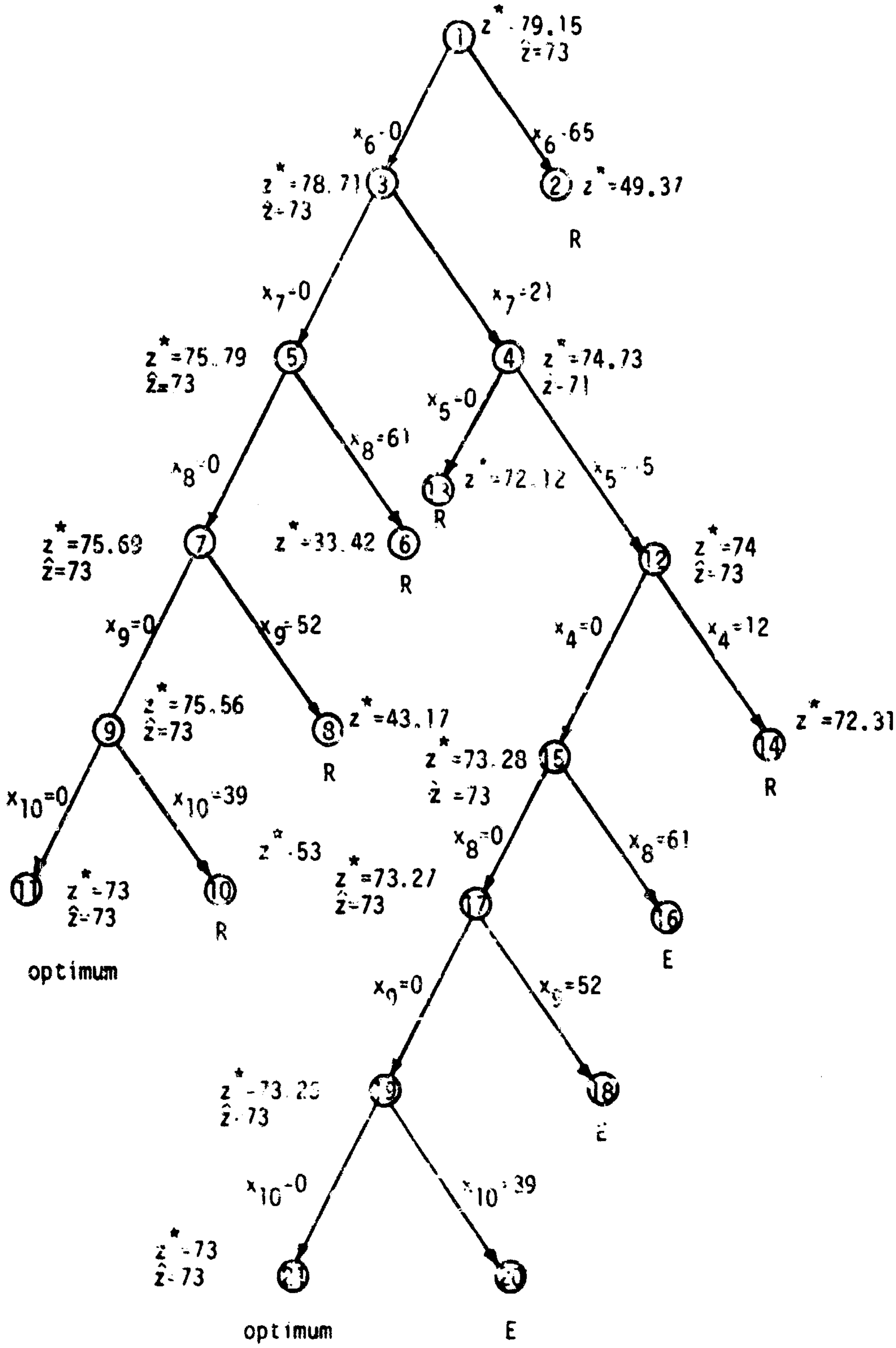


FIG. 4-2

4.5 A PARAMETRIC BRANCH AND BOUND ALGORITHM FOR THE KNAPSACK PROBLEM

This section is devoted to the derivation of an alternative algorithm for the solution of the knapsack problem. This method, as will be shown, is a special case of the branch and bound algorithm of Chapter II, with a more elaborate branching mechanism. The algorithm has certain features similar to the so-called *implicit enumeration* methods [7], [8], [9]. The most important characteristic of the method studied here is that it possesses the special property that the final tree configuration contains the optimal solutions for all similar knapsack problems with larger budgets than the one utilized to generate the tree. The branching part of the algorithm is similar to the one employed by Ichbiah [3] on a network connectivity analysis problem which guarantees the parametric properties of the resulting algorithm.

Let us rewrite our bivalent linear programming formulation of the knapsack problem where optimal $\underline{x}^0$ and z^0 are sought as to

$$Q : \text{Maximize} \quad z(B) = \sum_{i=1}^{m} c_i x_i$$

$$\text{Subject to} \quad \sum_{i=1}^{m} x_i \le B$$

$$0 \le x_i \le a_i \quad , \forall_i$$

$$x_i = 0 \text{ or } a_i \quad , \forall_i$$

Here $z^0(B)$ is the *knapsack function* corresponding to a specific value B of the budget. Let us define the sets S_1, T_1 and Ω_1 as follows:

$$S_1 = \{x_i \,/\, 0 \le x_i \le a_i\} \tag{4.28}$$

$$T_1 = \{x_i \,/\, \sum_{i=1}^{m} x_i \le B;\ x_i = 0 \text{ or } a_i\} \tag{4.29}$$

$$\Omega_1 = S_1 \cap T_1 = T_1 \tag{4.30}$$

The set S_1 is closed and bounded in E^m; T_1 is a nonempty and finite set for $B \geq 0$ and thus Ω_1 is finite.

Branching Operation. Given a certain node ℓ of the solution tree with associated sets Ω_ℓ and s_ℓ, the branching is defined by the intersection of the sets

$$v_{\ell,r} = \{x_i \ / \ x_k = a_k\} \tag{4.31}$$

$$v_{\ell,r+1} = \{x_i \ / \ x_k = 0\} \tag{4.32}$$

for a given $i = k$. The sets thus defined satisfy the sufficient conditions of Theorem 2.1, as may be easily verified. Also, since at each branching operation one variable is set to its only possible values and the number of variables is finite, the complete enumeration of solutions and thus the finiteness of the algorithm is guaranteed.

Auxiliary Problem. Associated with a newly-generated node ℓ of the solution tree, a continuous problem derived from Q must be solved. Denoting by I_0, I_a, and $\bar{I}$ the sets of variables fixed at a zero level, fixed at the upperbound, and free, problem A_ℓ takes the trivial form:

$$A_\ell : \text{Maximize} \quad z(\ell) = \sum_{i=1}^{m} c_i \, x_i$$

$$\text{Subject to} \quad 0 \leq x_i \leq a_i \qquad , i \in \bar{I}$$

$$x_i = 0 \qquad , i \in I_0$$

$$x_i = a_i \qquad , i \in I_a$$

Since we are maximizing over the set of free variables, bounded from above and since $c_i \geq 0$ is assumed, this linear program has as optimal solution:

$$x_r^* = \begin{cases} 0 & , \text{if } r \in I_0 \\ a_r & , \text{if } r \in I_a \cup T \end{cases} \tag{4.33}$$

Note that A_ℓ possesses always an optimal solution and therefore the exclusion operation, as defined in Chapter II, will never be applicable to this problem.

Termination Rule Rejection operations will not be used in this approach, hence the termination rule becomes the following as may be easily verified: terminate whenever the solution $\underline{x}^*$ to the auxiliary problem of the current bounded node is feasible, (i.e., $\underline{x}^* \in \Omega_1$).

LEMMA 4.2 *A solution $\underline{x}^*$ to A_ℓ is feasible for Q if it also satisfies* $\sum_{i=1}^{m} x_i^* \leq B$.

Proof: If $\underline{x}^*$ is optimal for A_ℓ, then from (4.33) $x_i^* = 0$ or a_i, $\forall_i$ and if also $\sum_{i=1}^{m} x_i^* \leq B$, then from (4.29) $\underline{x}^* \in T_1$ and from (4.30), $\underline{x}^* \in \Omega_1$.

COROLLARY 4.1 *If $\underline{x}^*$ is optimal for A_ℓ with $\sum_{i \in I_a} x_i^* > B$, the set S_ℓ associated with node ℓ does not contain a feasible solution to Q.*

From the above corollary, a node r of the tree for which $\sum_{i \in I_a} x_i^* > B$ may be excluded. Observe however that for a larger value of the budget the condition of the corollary may not be satisfied and the branch would not be deleted at that step.

Up to here the development parallels the additive algorithm of Balas as implemented by Geoffrion [8], with Corollary 3.1 providing the first rejection rule of Balas. From the above discussion it has been shown that Balas' enumerative algorithm may be interpreted as a branch and bound algorithm of the Land and Doig type

However, from here on our approach diverges from [8] in order to provide the parametric feature of the algorithm.

Fixed variable selection When a branching operation is to be performed based on the currently bounded node ℓ, one of the free variables must be selected to be fixed at its two possible values. We shall select the following criterion that will render "deterministic"* the generation of the branch and bound tree: Select for branching x_s, $s \in I$, such that

$$f_s = \min_{i \in I} [f_i] \tag{4.34}$$

If there is a tie, select the variable with greatest index. With this selection rule we create the directed arcs (ℓ,ℓ') and (ℓ,r). For the new node ℓ', $x_s = a_s$, (project s is accepted), the optimal solution to the auxiliary problem A_{ℓ}' would be the same as the solution to A_{ℓ}; furthermore, that solution is not feasible for problem Q.† Thus node ℓ' will be denoted a *pseudo-node* and no extra computation will be necessary whenever such a node is generated. As for node r, the solution to the auxiliary problem A_r has a value $z^*(r) \leq z^*(\ell)$ where $[z^*(\ell) - z^*(r)]$ is the smallest decrease possible in the objective function since, by (4.34), the variable fixed to zero is the one that has the minimum payoff of the set of free variables.

Whenever a pseudo-node ℓ' is generated from bounded node ℓ, then $z^*(\ell') = z^*(\ell)$ and the next bounded node of the solution tree would obviously be ℓ'. Accordingly, whenever branching takes place from ℓ to ℓ', an extra branch from ℓ' will also take place. This additional branch, performed by fixing to zero a new variable selected according to (4.34) will permit us to "look-ahead" on the solution tree. We remark that the branch from ℓ' fixing the selected variable to its upper bound remains to be performed and the algorithm must provide for its convenient generation.

---

* In the sense that any run of the problem would produce the same tree.

† If it was, ℓ being the bounded node, it would be an optimal solution.

4.6 STATEMENT OF THE ALGORITHM

Having satisfied the assumptions of the algorithm of Chapter II, and having indicated the branching mechanism to be employed, we proceed to state the *parametric branch and bound* algorithm.

STEP 1. Set $i = 1$. Generate node 1 by solving A_1 according to (4.34). Let z^*, $\underline{x}^*$ be the optimal solution and define $B^* = \sum_{i=1}^{m} x_i^*$. If $B^* \leq B$, stop; the solution is optimal. Otherwise, bound node 1 with $U_1 = z^*$. Set $i = i + 1$ and go to step i.

STEP i

A.1. BRANCH. Branch from bounded node ℓ. Select the variable x_s according to (4.34). Create pseudo-node ℓ' and node r, and directed arcs (ℓ,ℓ') and (ℓ,r). Solve problem A_r with $x_s = 0$. Set $x_s = a_s$ for pseudo-node ℓ'.

BRANCH AHEAD. Branch from pseudo node ℓ'. Select a new variable x_t according to (4.34). Create node $r + 1$ and directed arc $(\ell', r + 1)$. Solve problem A_{r+1} with $x_t = 0$. If the unique predecessor ℓ_0 of bounded node ℓ is a pseudo-node, go to A.2; otherwise go to B.

A.2. BRANCH. Branch from pseudo-node ℓ_0. Let x_p be the variable fixed to zero associated with the arc (ℓ_0,ℓ). Create pseudo-node ℓ_0' and directed arc (ℓ_0,ℓ_0'). Set $x_p = a_p$ for pseudo-node ℓ_0'.

BRANCH AHEAD. Branch from pseudo-node ℓ_0'. The variable to be fixed is x_t. Create node $r + 2$ and directed arc $(\ell_0', r + 2)$. Solve problem A_{r+2} with $x_t = 0$. Go to B.

B. BOUND. Select node ℓ such that $z^*(\ell) = \max_k \{z^*(k)\}$ for current terminal nodes. If $B^* \leq B$ for node ℓ, stop; the solution associated with node ℓ is optimal. Otherwise, upperbound node ℓ with $U_i = z^*(\ell)$, set $i = i + 1$ and go to step i.

The solution tree generated for a knapsack function with $B = B_0$ contains the optimal solutions (projects accepted as well as maximum payoff) for all knapsack functions with $B > B_0$, as is shown below. In this sense, the algorithm is parametric. By setting $B_0 = 0$ and applying the algorithm, a table may be constructed with the optimal solutions to problem Q for any non-negative value of the budgetary ceiling.

LEMMA 4.3. *Let $z^0(B)$ and $z^0(B_0)$ be the knapsack functions for two values of the budget, $B \geq B_0$. Then $z^0(B) \geq z^0(B_0)$.*

Proof: Assume $z^0(B) < z^0(B_0)$ and let $\underline{x}^0$ be the optimal solution for the budget B_0. Since $B \geq B_0$, then $\underline{x}^0$ is also feasible for the budget value B, hence $z(B) = z^0(B_0)$ for $\underline{x}^0$, a contradiction. Therefore, $z^0(B) \geq z^0(B_0)$.

THEOREM 4.3 *Let $T_0 = [N_0, A_0]$ be the final tree generated by determining the optimal solution to Q for $B = B_0$. Then for any $B \geq B_0$ there exists a node $k \in N_0$ such that the solution to the auxiliary problem A_j, say z^*, $\underline{x}^*$, constitutes an optimal solution for the corresponding knapsack problem.*

Proof: Given $B \geq B_0$, the application of the algorithm would generate a tree $T = [N, A]$ for which at least one node $k \in N$ corresponds to the optimal solution. The tree T is a *subgraph* of T_0, that is, $N \subseteq N_0$ and for all $(i, j) \in A$, also $(i, j) \in A_0$. In effect, since the algorithm is deterministic, the nodes of T_0 and T, generated in the same order, will have the same associated solutions to the auxiliary problems A_ℓ. It remains to be proved that when generating T, the optimal node k corresponds to a certain node of the generated portion of T_0. Assume the contrary, $k \notin N_0$; then the solution to A_k gives $z^*(B) < z^0(B_0)$, since any additional branching from T_0 reduces the upperbound value $z^0(B^0)$. From Lemma 4.3 this is a contradiction, and the solution for $B \geq B_0$ is not in the ungenerated portion of T_0, hence $k \in N_0$. It also follows that T is a subgraph of T_0.

We now give the following rule for retrieving the optimal node k. The set of bounded nodes at each iteration of the algorithm for $B = B_0$, form a sequence $[1, \ell_2, \ell_3, \ldots, \ell_n]$ where ℓ_i, $i \in N_0$ is the bounded node used for branching at iteration i and ℓ_n is the optimal node for $B = B_0$. Associated with each ℓ_i there are two numbers, $z^*(\ell_i)$ and $B^*(\ell_i)$. Due to the way the branch and bound algorithm has been developed, the $z^*(\ell_i)$ values constitute a non-increasing sequence. Therefore, for a given value $B \geq B_0$ it suffices to retrieve the first node for which $B^* \leq B$.

4.7 SOLUTION OF AN EXAMPLE PROBLEM

As an example consider the following problem involving five projects with payoffs and capital outlays as indicated in Table 4-3. Table 4-4 contains the necessary information for each iteration of the method. The budgetary ceiling considered is B = 10.

Project No.	f_i	a_i
1	6	3
2	4	5
3	3	6
4	2	4
5	1	2

TABLE 4-3

Note that a feasible solution is obtained at iteration 4 when node 6 is generated, but not until iteration 7 can it be bounded. Note also that at iteration 7 (when branching from node 13), all variables for the pseudo-node 13' are fixed, its B^* value is greater than the budget, and therefore it may be eliminated from further consideration. The same may be said of node 15.

The optimal solution for B = 10 is then obtained from node 6: accept projects 1, 2 and 5 with $z^0(10) = 11$

From Table 4-4 any optimal solution for $B > 10$ may be obtained. It suffices to search down the column of B^* and read off the solution from the row for which the first B^* less than or equal to the budget of

Step i	Terminal nodes	B^* for bounded node	Upper bound U_i
1	1*	20	16
2	2*, 3	18	15
3	3*, 4 , 5	16	14
4	4*, 5 , 6, 7 , 8	14	13
5	5 , 6 , 7, 8*, 9, 10	14	13
6	5 , 6 , 7, 9 , 10, 11, 12, 13*	15	12
7	5*, 6 , 7, 9 , 10, 11, 12, 14, 15	12	12
8	6*, 7, 9,10,11,12,14, 15, 16, 17	10	11

* bounded node

TABLE 4-4

interest is encountered. For $B = 14$ the fourth iteration provides an optimum, with $z^0(14) = 13$ associated with the bounded node 4, namely: accept projects 1, 2, and 3. An alternative optima is given by node 8: accept projects 1, 2, 4 and 5.

In Fig. 4-4, the optimal solutions for $B \geq 10$ are indicated graphically. Fig 4-3 contains the required solution tree.

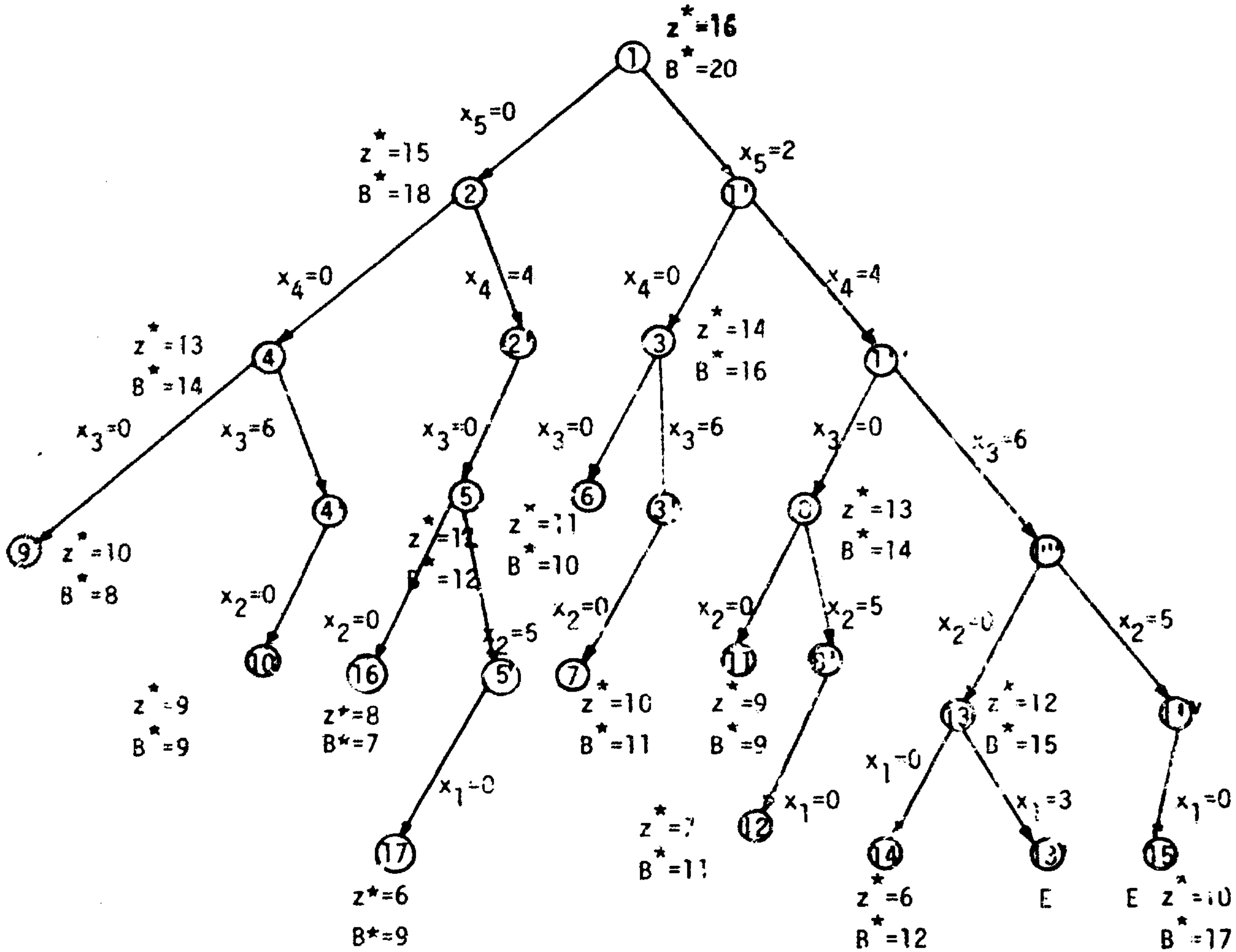


FIG. 4-3 The Solution Tree

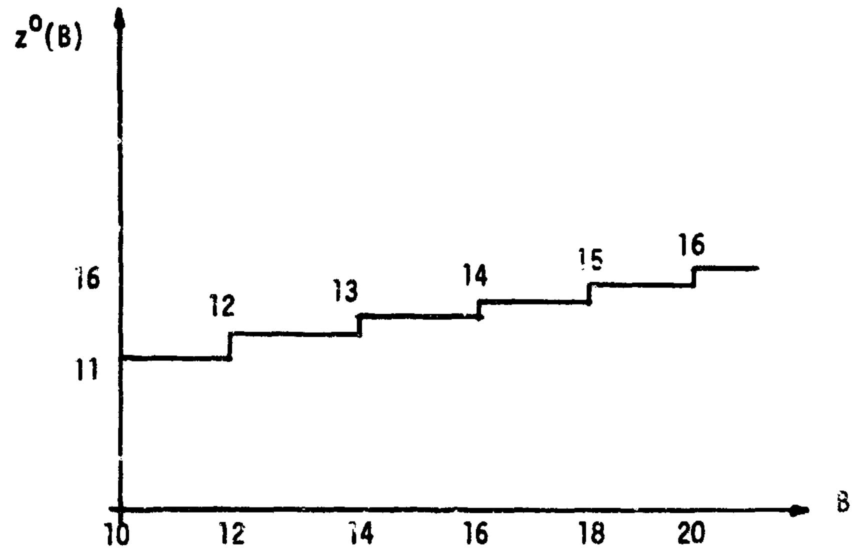


FIG. 4-4 Knapsack Function Values

4.8 SUMMARY

In this chapter we have addressed ourselves to the solution of the integer programming problem known as the knapsack problem. We have obtained the solutions to the dual problems, both in the linear and the discrete case, and have discussed the natural economic interpretation that may be drawn from such solutions. We have used discrete programming duality theory to justify Weingartner's approach to calculating dual prices on the primal resources.

The branch and bound algorithm developed for the multidimensional knapsack problem in Chapter III has been applied to the single period capital rationing problem with the consequent simplifications indicated. Finally, a parametric branch and bound algorithm has been derived which provides for sensitivity analysis studies of the optimal solution for variations of the budgetary ceiling within a certain specified range.

In the following chapter, formulation and solution techniques will be provided for the problem of capital allocation to independent projects, where the investment decisions may be deferred to later periods.

4.9 NOTES TO CHAPTER IV

[1] Gilmore, P. C., and R. E. Gomory, "The Theory and Computation of Knapsack Functions", *Operations Research*, Nov. - Dec., 1966, pp. 1045-1074.

[2] Balas, E., "An Additive Algorithm for Solving Linear Programs with 0-1 Variables", *Operations Research*, July - August, 1965, pp. 517-549.

[3] Ichbiah, J. D., *Connectivity Analysis and Branch and Bound Methods*, Ph.D. Thesis, Massachusetts Institute of Technology, Department of Civil Engineering, August 1967.

[4] Lorie, J. H. and L. J. Savage, "Three Problems in Rationing Capital", *The Journal of Business*, Vol. XXVIII, October 1955, pp. 229-239.

[5] Balas, E., "Duality in Discrete Programming", Technical Report No. 67-5, Operations Research House, Stanford University, August 1967.

[6] Weingartner, H. M., *Mathematical Programming and the Analysis of Capital Budgeting Problems*, Prentice-Hall, 1963.

[7] Glover, F., "A Multiphase-Dual Algorithm for the Zero-One Integer Programming Problem", *Operations Research*, Nov. - Dec., 1965, pp. 879-919.

[8] Geoffrion, A. M., "Integer Programming by Implicit Enumeration and Balas' Method", *SIAM Review*, Vol. 9, No. 2, 1967, pp. 178-190.

[9] Kolesar, P. J., "A Branch and Bound Algorithm for the Knapsack Problem", *Management Science*, Vol. 13, May 1967, pp. 723-735.

CHAPTER V

MULTISTAGE RESOURCE ALLOCATION PROBLEMS

5.1 INTRODUCTION

In the preceding chapter we studied the problem of optimally allocating funds among competing alternatives, each requiring investment in a number of time periods and with a fixed horizon, subject to budgetary constraints in each period. An effective solution technique was developed. The purpose of this chapter is to study the problem of capital allocation among independent projects that arises in various planning contexts, where the projects require fixed outlays in one time period, but the decision of accepting them may be deferred to a later period. An expected net benefit (financial and social) is assumed to be known for each project, and the figure of merit adopted is to maximize the total benefit.

We begin by formulating the multistage capital allocation problem as a (0-1) integer program. A special case is then considered where capital outlays for all projects do not vary over time and any infationary effects are taken into consideration by modifying the budget ceilings accordingly. The problem is referred to as the *multi-knapsack* problem. A suitable transformation is performed to obtain an equivalent model which is interpreted as an analysis-synthesis problem on a bipartite network. The general branch and bound algorithm of Chapter II is then adapted to provide a convenient solution method.

Finally, a second formulation for the multi-knapsack problem is derived for which a branch and bound algorithm is proposed. In this case, the solution of the auxiliary problem associated with each node of the solution tree may be obtained by inspection

5.2 MULTISTAGE CAPITAL ALLOCATION MODELS

Consider a certain government agency confronted with the problem of allocating a multistaged budget with ceilings $B_j \geq 0$, $j=1,\ldots,n$, among a set of m independent projects each requiring a unique capital outlay. Let f_{ij} be the expected payoff, determined from a linear utility function, of investing in project i at time period j. Further, let $a_{ij} \geq 0$ be the capital outlay required for project i if selected for investment at time period j, and y_{ij} the associated decision variable that may take on the values 0 or 1 depending on whether project i is rejected or accepted for investment in period j. Then the problem of selecting a set of projects for investment so that the total utility over time is maximized, while satisfying the funding dependencies represented by the budgetary ceilings, may be formulated as

$$P_1 : \text{Maximize} \quad z = \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij}\, y_{ij}$$

$$\text{Subject to} \quad \sum_{i=1}^{m} a_{ij}\, y_{ij} \leq B_j \quad , j=1,\ldots,m \qquad (5.1)$$

$$\sum_{j=1}^{n} y_{ij} \leq 1 \quad , i=1,\ldots,n \qquad (5.2)$$

$$y_{ij} \geq 0 \quad , \forall_{i,j} \qquad (5.3)$$

$$y_{ij} \text{ integer} \quad , \forall_{i,j} \qquad (5.4)$$

where constraints (5.1) express the budget limitations; and where constraints (5.2) serve the double task of guaranteeing that project i, if accepted at any one period, incurs a unique capital outlay and, concurrently with (5.3) and (5.4), that the variable y_{ij} may only take the values 0 or 1.

Problem P_1 is an all-integer linear programming problem with (m+n) constraints and mxn integer variables for which a feasible solution and a lower bound are immediately available, corresponding to the *status quo* or reject-all-projects policy. Observe also that by collapsing the number of time periods to one, the index j may be dropped and thus problem P_1 becomes, as expected, the (0-1) knapsack problem studied in Chapter IV.

The solution to problem P_1 may be attained by direct application of the branch and bound algorithm of Chapter II. In this case, when constraint (5.4) is relaxed the problem becomes the well-known *weighted distribution problem*, sometimes called the *generalized transportation problem*. Therefore the primal method of Dantzig [1] or the dual method of Balas [2] could in principle be used as subalgorithms for solution of the auxiliary problems associated with each node of the branch and bound tree.

We shall, however, address ourselves to the special case of P_1 in which the capital outlay for each project remains the same regardless of the period chosen for investment; this amounts to assuming non-inflationary costs throughout the horizon of interest. The problem thus obtained has wide application to various resource allocation problems.

5.3 THE MULTI-KNAPSACK PROBLEM. AN EQUIVALENT MODEL

Problem P_1 with the additional condition that the a_{ij} are the same for all j, which we shall call the *multi-knapsack* problem, is a generalization of Dantzig's knapsack problem. It can be expressed as follows: determine the optimum packing of a set of m articles into a set of n knapsacks, given the desirability f_{ij} of each item for each knapsack, the weight a_i of each item, and the maximum weight B_j that each knapsack is allowed to carry.

This problem is of special importance in the operation of transportation terminals, where optimal cargo loading into vehicles of varying capacities is desired. Also, it arises in a computer environment, where programs or files of a given size are competing for non-connected fixed size data storage pools. These are but two examples of optimization problems that can be modelled as multi-knapsack problems.

We shall study this problem in terms of an alternative model equivalent to problem P_1. With this equivalent model, the problem will be interpreted as a network analysis-synthesis problem which resembles the plant location problem with fixed charges on links with positive flows. A branch and bound algorithm for the solution of this problem will also be developed.

Consider problem P_1, with a_j replaced by a_i as discussed above. Replacing the set of variables y_{ij} by the new variables $x_{ij} = a_i y_{ij}$, and letting $c_{ij} = \frac{f_{ij}}{a_i}$, we obtain the following equivalent program:

$$P : \text{Maximize} \quad z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}$$

$$\text{Subject to} \quad \sum_{i=1}^{m} x_{ij} \le B_j \quad , \ j=1,\ldots,n \tag{5.5}$$

$$\sum_{j=1}^{n} x_{ij} \le a_i \quad , \ i=1,\ldots,m \tag{5.6}$$

$$x_{ij} = 0 \text{ or } a_i \quad , \ \forall\ i,j \tag{5.7}$$

Problem P is a discrete bivalent linear programming problem where the x_{ij} represent capital outlays to be determined and the c_{ij} represent the utility of project i at time period j, per unit of outlay required.

Let us interpret the set $N_1 = [i \ / \ i = 1,\ldots,m]$ as a set of "origin" nodes, $N_2 = [j \ / \ j = 1,\ldots,n]$ as a set of "destination" nodes, and the set A of ordered pairs (i,j) as arcs joining nodes i and j. Furthermore, let a_i be the "demand" or input of node $i \in N_1$ and B_j the "demand" or output of node $j \in N_2$, and interpret x_{ij} as flow on the arc (i,j). We may then associate a bipartite network $G = [N_1, N_2, A]$ to problem P; or better still an equivalent network with a single source and a single sink. This latter step may be achieved by adding artificial nodes s and t, and artificial arcs (s,i), $\forall\ i \in N_1$ and (j,t), $\forall\ j \in N_2$, with the following associated values: $c_{si} = 0$, $u_{si} = a_i$ and $c_{jt} = 0$, $u_{jt} = B_j$; where u_{si} and u_{jt} denote respectively the upper bounds on the arcs (s,i) and (j,t). The associated network is shown in Fig. 5-1. The first number on each arc represents cost and the second represents arc capacity

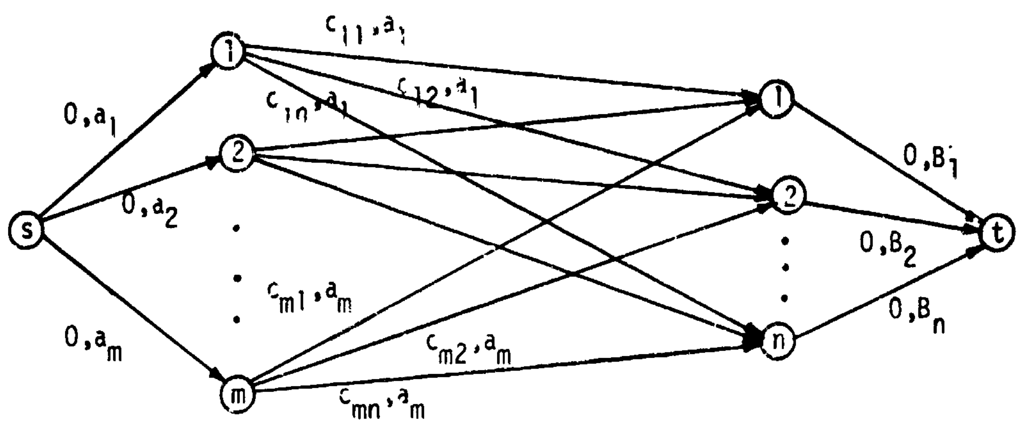


FIG. 5-1

The multi-knapsack problem therefore may be expressed in terms of network flow theory as an *analysis-synthesis* problem: find the maximum flow from s to t that maximizes cost on the network of Figure 4-1, subject to the restriction that arcs (i,j), $i \in N_1$, $j \in N_2$ are either not used or saturated; and find as well its distribution pattern. Note that since the upper bounds on arcs (s,i) are a_i, if project r is accepted only one arc (r,j) will be activated.

The optimal solution determines which projects will be accepted for investment (not all arcs (s,i) need be saturated); and it determines to which destination node they will be assigned, thus completing the synthesis portion of the problem.

Observe that problem P differs from a standard transportation problem with surplus and deficit in that each origin, if used at all for shipping, must supply a single destination node. (It also differs in that P is a maximization problem) If, however, the constraint (5.7) is relaxed so that the x_{ij} are simply restricted to be non-negative, the resulting program indeed corresponds to a transportation problem with surplus and deficit. This fact will be employed in proposing a subalgorithm for solution of the auxiliary problem during our development of the branch and bound solution method in Section 5.4.

The model as presented in formulation P may be extended to consider more flexible cases which might add relevance to the problem or be adapted to a more realistic situation. When budget deferrals are allowed, thus transferring unused capital to a later period, the P formulation must be modified, as indicated below, to account for such flexibility. Let s_j be the unused budget, if any, at time period j. Then it suffices to modify the constraints (5.5), replacing them instead with the following one:

$$\sum_{i=1}^{m} x_{ij} + s_j - s_{j-1} = B_j \ , \ j=1,\ldots,n \tag{5.8}$$

where $s_0 \equiv 0$. We remark that the coefficient of s_{j-1} is one, hence no present worth factor has been considered in (5.8) and thus the slacks simply represent idle costs. Other generalizations may be made, such as lending and borrowing in the capital market. (See for example, [3], Chapters 8 and 9.)

The network representation for problem P with (5.8) instead of (5.5) would correspond to the one of Fig. 5-1 with additional directed arcs: (n, n-1),...,(2,1). These arcs are uncapacitated (unless deferred expenditure is specifically bounded) and have zero costs.

5.4 DEVELOPMENT OF A SOLUTION METHOD

We shall derive a branch and bound method for the solution of problem P by utilizing the same terminology and notation employed in Chapter II. We begin by defining the sets S_1, T_1 and Ω_1 as follows:

$$S_1 = \{\underline{x} \ / \ \sum_{i=1}^{m} x_{ij} \le B_j, \ \sum_{j=1}^{n} x_{ij} \le a_i, \ x_{ij} \ge 0, \ \forall \ (i,j)\}$$

$$T_1 = \{\underline{x} \ / \ x_{ij} = 0 \text{ or } a_i\}$$

$$\Omega_1 = S_1 \cap T_1$$

We observe that S_1 is a closed and bounded convex set, since it is defined as the intersection of a finite number of closed convex half spaces; furthermore, any vector $\underline{x}$ with $x_{ij} < 0$, $\forall$ i,j constitutes a lower bound, and, for example, $\underline{x}$ with $x_{ij} = B_j$ constitutes an upper bound*. Also, T_1 is a finite non-empty set determined by the $2^{m\,n}$ vertices of the rectangular polyhedron defined by $0 \le x_{ij} \le a_i$, $\forall$ i,j. Finally, Ω_1, defined as the intersection of S_1 and T_1, is finite, since T_1 is finite, and non-empty, since at least $\underline{x} = 0$ belongs to the intersection Ω_1. Since Ω_1 is non-empty, an optimal solution to problem P always exists.

Branching operation. Given a certain node ℓ of the solution tree with associated sets Ω_ℓ and S_ℓ, the branching is defined by their intersection with the sets

$$V_{\ell,r} = \{x_{ij} \,/\, x_{st} = 0\}$$

$$V_{\ell,r+1} = \{x_{ij} \,/\, x_{st} = a_s\}$$

for a given i = s and j = t. The sets thus defined satisfy the first sufficiency condition of Theorem 2.1, that is:

$$V_{\ell,\,r} \cap V_{\ell,r+1} = \{x_{ij} \,/\, x_{st} = 0,\ x_{st} = a_s\} = \Phi.$$

Also, since Ω_ℓ is a subset of Ω_1 (by the branching operation) then from (5.7) the x_{st} components of the vector elements of Ω_ℓ must be either 0 or a_s. Then $\Omega_\ell \cap (V_{\ell,r} \cap V_{\ell,r+1}) = \Omega_\ell \cap \{x_{ij} \,/\, x_{st} = 0 \text{ or } x_{st} = a_s\} = \Omega_\ell$ and the second condition for sufficiency of Theorem 2.1 is also satisfied.

Finally, since at each branching operation one variable is fixed to each one of its possible values, the finiteness of the number of variables assures that only a finite number of branching operations are required before total enumeration of the elements of Ω_1 is accomplished.

---

* Obviously, no capital outlay may exceed the sum of all the budgets; if so, it may be ruled out of the problem.

5.5 THE AUXILIARY PROBLEM AND ITS SUBALGORITHM

At each iteration of the branch and bound algorithm, associated with each newly-generated node ℓ of the solution tree, a continuous *auxiliary problem* A_ℓ derived from P must be solved.

Denote by $I_0 \subseteq A$ the subset of arcs $(i,j) \in A$ of the network G for which $x_{ij} = 0$ (i.e., investment on project i rejected at period j): by $I_a \subseteq A$ the subset of arcs with $x_{ij} = a_i$ (i.e., project accepted in period j); and by $\bar{I}$, the set of "free" arcs. Then the auxiliary problem A_ℓ takes the form

$$A_\ell : \text{Maximize} \quad z(\ell) = \sum_{i=1}^{m} \sum_{j=1}^{m} c_{ij} x_{ij}$$

$$\text{Subject to} \quad \sum_{i=1}^{m} x_{ij} \leq B_j \quad , j=1,\dots,n$$

$$\sum_{j=1}^{n} x_{ij} \leq a_i \quad , i=1,\dots,m$$

$$x_{ij} = 0 \quad , (i,j) \in I_0$$

$$x_{ij} = a_i \quad , (i,j) \in I_a$$

$$x_{ij} \geq 0 \quad , (i,j) \in \bar{I}$$

This is a transportation type linear program in inequality form with some prohibited routes* and where maximization is sought. Therefore, this problem may be solved by any available transportation algorithm. In particular, the generalized primal-dual algorithm of Fulkerson [4] is perfectly suited for this problem. Indeed, at each node r of the branch and bound tree, the solution to the auxiliary problem of the unique

---

* The arcs $(i,j) \in A$ for which $x_{ij} = a_i$ may be interpreted as prohibited routes if a_i is first subtracted from the corresponding nodes i and j

predecessor node ℓ of r may be used as a starting flow. The out-of-kilter algorithm will then reoptimize this flow according to the status of sets I_0 and I_a associated with node r. If r corresponds to the branch $x_{ij} = 0$, it suffices to set the lower and upper bounds on that arc equal to zero. If, on the other hand, the branch corresponds to $x_{ij} = a_i$, the lower and upper bounds will be set equal to a_i.

Observe also that if an optimal solution to A_r has been obtained, given by

$$x^*_{ij} = \begin{cases} 0 & , \text{ if } (i,j) \in I_0 \\ a_i & , \text{ if } (i,j) \in I_a \\ 0 \leq x^*_{ij} \leq a_i & , \text{ if } (i,j) \in \bar{I} \end{cases}$$

then a feasible solution to P, provided by a rounding operation on $\underline{x}^*$, is:

$$\hat{x}_{ij} = \begin{cases} x^*_{ij} & , \text{ if } x^*_{ij} = 0 \text{ or } a_i \\ 0 & , \text{ if } 0 < x^*_{ij} < a_i \end{cases}$$

This simple operation permits the use of the double bounding technique as well as the rejection operation of the branch and bound algorithm, which we proceed to enunciate:

STEP 1. Set $i = 1$ and create node 1. Solve A_1. If $\underline{x}^*$ is such that all $x^*_{ij} = 0$ or a_i, stop; the solution is optimal. If at least one $x^*_{ij} \neq 0$ or a_i, bound node 1 with $U_1 = z^*(1)$. Round node one to obtain $\hat{\underline{x}}$, $\hat{z}(1)$. Set $L_1 = \hat{z}(1)$. Here $U_1 > L_1$. Set $i = i + 1$ and go to step i.

STEP i. a) BRANCH. Branch from bounded node ℓ. Create nodes r and $r + 1$ and directed arcs (ℓ,r) and $(\ell,r+1)$. Select any x_{ij} such that $0 < x^*_{ij} < a_i$ for node ℓ, and branch with $x_{ij} = 0$ and $x_{ij} = a_i$. Solve A_r using the subalgorithm and then A_{r+1} based on the solution to A_r. If A_r or A_{r+1} are infeasible, exclude them from further consideration.

b) ROUND. Round nodes r and r + 1 to obtain $\hat{z}_r$ and $\hat{z}_{r+1}$.

c.1) BOUND FROM BELOW. Set $L_i = \max\,[L_{i-1}, \hat{z}(r), \hat{z}(r+1)]$. Reject all nodes with $z^* < L_i$.

c.2) BOUND FROM ABOVE. Select node ℓ such that $z^*(\ell) = \max_k [z^*(k)]$, for current terminal nodes. Upper bound node ℓ with $U_i = z^*(\ell)$. If $L_i = U_i$, stop; the feasible solution that provides the lower bound is optimal. Otherwise $L_i < U_i$. Set i = i +1 and go to step i.

5.6 ALTERNATIVE FORMULATION FOR THE MULTI-KNAPSACK PROBLEM

In this section we present an alternative formulation for the multi-knapsack problem which permits the solution of the auxiliary problems of the branch and bound tree by inspection.

Associated with each project $i \in N_1$, we introduce a decision variable y_i restricted to take the values 0 or 1, which indicate rejection or acceptance of the project i, respectively.

Then problem P may be formulated as follows:

$$P_2 : \text{Maximize} \quad z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}\, x_{ij} \tag{5.9}$$

$$\text{Subject to} \quad \sum_{i=1}^{m} x_{ij} \le B_j \quad , j=1,\ldots,n \tag{5.10}$$

$$\sum_{j=1}^{n} x_{ij} = a_i\, y_i \quad , i=1,\ldots,m \tag{5.11}$$

$$x_{ij} = 0 \text{ or } a_i \quad , \forall\, i, j \tag{5.12}$$

$$y_i = 0 \text{ or } 1 \quad , \forall\, i \tag{5.13}$$

Here constraints (5.11) and (5.13) establish the fact that, if project i is accepted, $y_i = 1$, the total outlay over all time periods is a_i. In addition, constraints (5.12) guarantee that the outlay a_i will be disbursed in only one of the stages considered.

We may express P_2 in a more convenient form for developing a solution method, as follows:

$$P_2 : \text{Maximize} \quad z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{5.14}$$

$$\text{Subject to} \quad \sum_{i=1}^{m} x_{ij} \le B_j \quad , j=1,\dots,n \tag{5.15}$$

$$0 \le x_{ij} \le a_i \quad , \forall i, j \tag{5.16}$$

$$\sum_{j=1}^{n} x_{ij} / a_i = 0 \text{ or } 1 \ , \forall i \tag{5.17}$$

$$x_{ij} = 0 \text{ or } a_i \quad , \forall i, j \tag{5.18}$$

We observe that, if the discrete constraints (5.17) and (5.18) are relaxed, the resulting linear program, (5.14) subject to (5.15) and (5.16), may be solved by inspection. Indeed, it is composed of n mutually independent linear programs, each one associated with one time period $j \in N_2$

Under the assumption of non-negative c_{ij}, and after ordering the c_{ij} for each $j \in N_2$ in decreasing order, the optimal solutions may be obtained by an expression analogous to (3.25)

The branch and bound algorithm may therefore be applied directly to problem P_2 above

At each step of the algorithm, one variable, not currently satisfying (5.18), is fixed to its possible values, thus defining the branching operation.

We remark that for a certain node of the solution tree the values of y_i calculated according to (5.17) may be greater than one; in fact, they may be as large as n, where n is the number of time periods.

Note also that a rounding operation may be performed at each iteration. However, if the solution of the auxiliary problem results in a certain project i with capital outlays a_i in various time periods, more than one feasible solution to P_2 may be obtained by application of the rounding operation.

When a comparison is made of the branch and bound algorithms developed in Section 5.5 and the one indicated here, we may point out the following: The first formulation P requires the solution of a network flow problem at each node, as opposed to the solution of n simple linear programs solved by inspection when the formulation P_2 is used. However, the second approach in general requires the search of a larger number of nodes before optimality is reached.

5.7 OPTIMAL ALLOCATION OF PROGRAMS TO PRIMARY MEMORY

We conclude this chapter with the formulation of a problem which is related to the multi-knapsack case and which arises in the context of allocating programs to primary memory in a computer system.

Consider a set of m items of size a_i, i = 1,..., m that are to be loaded into n knapsacks of capacity B_j, j = 1,..., n. We assume $\sum_{j=1}^{n} B_j \geq \sum_{i=1}^{m} a_i$. The problem is to assign items to knapsacks so that the minimum number of knapsacks is used. The use of each knapsack incurs a fixed cost f_j, and thus the total cost of using the knapsacks is to be minimized.

The problem may be formulated as follows:

$$P_3 : \text{Minimize} \quad z = \sum_{j=1}^{n} f_j y_j \tag{5.19}$$

$$\text{Subject to} \quad \sum_{i=1}^{m} x_{ij} \leq B_j \tag{5.20}$$

$$\sum_{j=1}^{n} x_{ij} = a_i \tag{5.21}$$

$$\sum_{i=1}^{m} \frac{x_{ij}}{a_i} \leq m\, y_j \tag{5.22}$$

$$y_j = 0 \text{ or } 1 \tag{5.23}$$

$$x_{ij} = 0 \text{ or } a_i \tag{5.24}$$

where the decision variable y_j is associated with each knapsack. Constraints (5.22) guarantee that no flow will occur from nodes $i \in N_1$ to node $j \in N_2$ if $y_j = 0$.

Again, problem P_3 may be solved by direct application of the branch and bound technique. We shall indicate here that if constraints (5.23) and (5.24) are relaxed, the resulting linear program may be solved by means of a network flow algorithm.

Indeed, since we are minimizing, the optimal solution to P_3 without discreteness constraints will necessarily satisfy (5.22) as a strict equality:

$$\sum_{i=1}^{m} \frac{x^o_{ij}}{a_i} = m\, y^o_j \Rightarrow y^o_j = \frac{1}{m} \sum_{i=1}^{m} \frac{x_{ij}}{a_i} \tag{5.25}$$

Substituting (5.25) in the objective function, the problem to be solved is:

$$\text{Minimize} \quad z = \frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{f_i}{a_i} x_{ij}$$

$$\text{Subject to} \quad \sum_{i=1}^{m} x_{ij} \leq B_j$$

$$\sum_j x_{ij} = a_i \quad , \quad x_{ij} \geq 0$$

which obviously corresponds to a transportation problem, and thus a network flow algorithm may be employed for its solution.

5.8 NOTES TO CHAPTER V

[1] Dantzig, G. B., *Linear Programming and Extensions*, Princeton University Press, 1963, Ch. 21.

[2] Balas, E., "The Dual Method for the Generalized Transportation Problem", *Management Science*, Vol. 12, March 1966, pp. 555-568.

[3] Weingartner, H. M., *Mathematical Programming and the Analysis of Capital Budgeting Problems*, Prentice-Hall, Inc., Englewood Cliffs, 1963.

[4] Fulkerson, D. R., "An Out-of-Kilter Method for Minimal Cost Flow Problems", *Journal Soc. Indust. Appl. Math.*, Vol. 9, 1961, pp. 18-27.

CHAPTER VI

THE MULTISTAGE NETWORK DESIGN PROBLEM

6.1 INTRODUCTION

In this chapter we shall propose a programming model aimed at determining an optimum transportation network development plan for a metropolitan area. The model to be derived synthesizes the best network configuration among a set of suggested improvements maintaining expenditures within expected future budget ceilings. The budgetary constraints, projected into the future from trend studies of past transportation expenditures, are assumed to be given up to a fixed horizon in a predetermined staging sequence. The model furthermore assumes the continuity of a stable technology over the entire period of interest.

The figure of merit selected for optimization is the total user cost over all time periods. The required input data are the expected origin-destination demands for all time periods over the existing network; the subset of existing links selected for capacity improvement with the corresponding capital requirement; and/or the construction costs of specific links to be added to the current network and their total improvement of capacity if selected for construction. Also required is the topology of the existing network, with link capacities and estimated users cost per link for all periods of interest.

The problem described is interpreted as a capital investment problem with dependent projects, where the desirability of combinations of projects will be reflected in the redistribution of flow volumes and therefore in a reduction of total users cost. The projects' interdependency relationships are taken into consideration by imbedding into the basic model a network flow distribution submodel. In this fashion, the multistaged network design problem is interpreted as a combined network flow and capital budgeting problem.

The various topics considered in this chapter are organized in the following manner: we begin by presenting a general overview of the urban transportation planning process with emphasis on the various classes of network improvement evaluation models, their characteristics and main drawbacks. Next, a discussion of the various levels of network improvements and a review of existing models for each type of improvement is presented. With this background material, the basic multistaged model is developed in terms of a highly-structured mixed-integer linear programming formulation. The structure of the model is then thoroughly analyzed in order to propose a convenient optimization technique for carring out its solution. The proposed solution procedure is the partitioning algorithm of Benders (cf. Appendix A) which fully utilizes the decomposable nature of the multi-stage problem.

6.2 THE URBAN TRANSPORTATION PLANNING PROCESS

The need for an integrated long-range transportation plan for metropolitan areas has been widely recognized by civil engineers, city planners, economists, sociologists, city officials, etc. in the postwar era, as a result of the explosive increase in the size and complexity of urban areas. The need for such a master plan has been officially endorsed by Congress in the Federal-Aid Highway act of 1962, which grants federal aid to urban areas of more than fifty thousand population, provided that their projects are based on "a continuing comprehensive transportation planning process...".

The planning process is primarily concerned with forecasting future demand for transportation in a certain study area, as well as planning transport facilities that provide a satisfactory level of service while maintaining the corresponding capital expenditures within expected future budget ceilings.

Comprehensive studies such as the Chicago Area Transportation Study (CATS), Penn-Jersey Transportation Study, etc., have been carried out from a systems viewpoint; the attempt to consider all the interacting elements that affect the demand for transportation, and to plan new facilities in the light of their interaction with the existing network (as opposed to making local decisions or accepting small-scale palliative solutions). For a general description of different issues akin to urban transportation studies see Moyer [1].

The relation of urban development to demand for transportation and the effect of new facilities upon demand patterns have been carefully identified. Past research has focussed on the characteristic steps of the planning process; quantitative approaches, such as use of mathematical models and techniques, have been suggested for each step of the planning process, and much experience and momentum has been gained from these studies.

Most transportation planning models are based on expediting the extrapolated trend of economical and environmental development.

Although the transportation studies carried on for various metropolitan areas had to treat different problems according to the specific areas of interest, they present virtually the same pattern in their solution approach. This pattern indicates the fundamental steps of the transportation planning process, which we proceed to enumerate.

i) Inventories for base year:

These consist of inventory, for a reference year T_b (base year), of the relevant factors that will affect the future demand for transportation. The inventories usually considered, mostly based on censuses, are listed below.

i land use inventory
ii population inventory
iii transportation inventory
iv trend of transportation expenditures

ii) Inventories for target year: These are developed by forecasting, for the *target year* T_t (usually a 20 or 25 year interval), the changes in the *base year* inventories.

This forecasting is usually attained, for each type of inventory, by means of prediction models of varying sophistication. Martin, Memmott and Bone [2] present an analysis and detailed description of various models often used in the planning process.

It is interesting to observe that these first two steps are invariably required for an integrated study of the development and improvement of any kind of facilities in a metropolitan area.

iii) Transportation Analysis:

Based on the demand for transportation at T_B and on both the base year and target year inventories, the future demand for transportation is forecast, and a master transportation plan is developed. This analysis constitutes the core of the transportation planning process and its accuracy will be a direct function of the accuracy and completeness of the forecasting analysis.

6.3 TRANSPORTATION ANALYSIS

We shall briefly mention the now well-established steps into which the transportation analysis phase is subdivided. For each step, well-developed models are available, and a substantial amount of research is currently underway seeking to verify and improve the accuracy of such models.

i) *Trip Generation.* The purpose of trip generation is to determine the number of trips starting (or ending) in a particular zone of the study area for specified future years.

ii) *Trip Distribution.* Trip distribution is the process of assigning destinations, by means of a distribution model such as the gravity model, to the trips generated in each zone of the study area.

iii) *Modal Split.* The modal split analysis is used to estimate the future breakdown of trips among the available transportation modes. The models most frequently employ multiple regression analysis, and are used to predict future modal split for the modified values of the input variables; this clearly implies that no major changes in transportation technology are expected during the period of interest.

The modal split phase assigns each future traffic demand by mode to the corresponding transportation network for that mode.

iv) *Traffic Assignment.* The objective of the traffic assignment is to determine flow patterns in specific transportation networks, where flows are associated with the different modes adopted in the planning process. This step, being of special interest for the present work, will be treated in more detail in forthcoming sections.

v) *Transportation System Evaluation*. The traffic assignment step is usually performed for each mode and for alternative transportation networks, with the main objective of obtaining substantial information on the relative performance of the alternatives. This will hopefully permit the rational selection of the transportation system that will best meet the future demand with a suitable level of service.

The evaluation of the various alternatives is usually done by standard techniques, such as cost-benefit analysis or rate-of-return method. In a forthcoming section this evaluation step will be analyzed and identified as a capital budgeting problem which can be systematically and quantitatively attacked

The output of the evaluation phase, possibly obtained after several iterative cycles of the total process, will be the desired long-range urban transportation plan for the area of interest

6.4 THE TRAFFIC ASSIGNMENT PROBLEM

In the context of transportation planning, the term traffic assignment means the determination of flow volumes on the links of a given transportation network, where volumes per unit of time are specified between each zonal pair in a set of origin-destination pairs. The traffic assignment permits the evaluation of the performance of network alternatives.

The question of how the flow distributes itself over the network constitutes one of the most important issues in transportation planning. Two different criteria, enunciated by Wardrop [3] and formalized in mathematical form by Beckman et al. [4] and Charnes and Cooper [5], have initiated the development of two major classes of traffic assignment models. These are generally given the titles of *descriptive* (predictive) and *normative* (prescriptive) models. Each Wardrop postulate suggests that the flow distributes itself over the network according to one of two contrasting extremal principles:

i) Postulate of equal travel times: for a flow assignment, the travel time between any two points on the network will be the same on all routes used and less than the travel time on any other path joining the same two points.

ii) Postulate of overall minimization: for an optimal flow assignment, the average travel time for all users of the network attains its minimum value.

Descriptive Traffic Assignment Models

This family of traffic assignment models is based on Wardrop's principle of equal travel times. The computer implementation of such models has acquired great momentum as a result of their use in transportation studies of major metropolitan areas during the early sixties. These programs implicitly use the game theory model of Charnes and Cooper, where all travelers seek to minimize their own travel time.

The flow distribution is achieved by iteratively assigning traffic from each origin node to all destinations according to current shortest path-routes. After completion of each iteration, the resultant travel times on links are updated according to their current loads and the origins will again take turns assigning portions of their flows.

The descriptive models used in different transportation studies present variations in their actual calculation, but they are all based on the principles indicated above. In [6] and [7] the reader will find a complete description and comparison of the various models in use today.

Normative Traffic Assignment Models

This class of models is based on Wardrop's postulate of overall minimization and on the traffic flow analysis of Beckman et al. and Charnes and Cooper: flows distribute themselves so as to minimize the total travel time in the system, as opposed to individual travel times.

This optimization problem has been formulated by Charnes and Cooper [5], for congested networks, as a non-linear programming problem; the non-linearity results from the fact that link travel times increase non-linearly with flow volumes. They further simplify their model by suggesting a piece-wise linearization of the travel time-volume relationship, accomplished by introducing multiple capacitated arcs with increasing travel times. The resulting model is a linear program known as the multicopy-cost-minimization network flow problem. This problem has been thoroughly analyzed and exploited by Pinnell and Satterly [8] and by Hershdorfer [9].

Jorgensen [10] has studied both classes of traffic assignment models, and shows that for the uncongested case (rural networks), both the descriptive and normative solutions give the same flow distribution pattern.

The actual computer implementation of normative models requires a linear programming routine capable of handling a potentially large number of constraints; or alternatively, with the additional capability of exploiting the highly-structured form of the model by conveniently decomposing the problem into more tractable subprograms.

6.5 TRANSPORTATION NETWORK IMPROVEMENTS

The main goal of the traffic assignment is to determine the level of service provided by a given network for a set of demands previously specified. When the demand expected for the target year is assigned to the network configuration of the basic year, it is likely that the latter will not provide a satisfactory level of performance. This condition will be reflected in the final assignment by an excessive number of links operating under congestion.

On the other hand, if new urban areas are expected to be developed by the target year, the network will have to be expanded to provide transportation facilities to these areas.

This situation clearly calls for a network improvement plan to meet the forecasted demand, making use of the limited capital resources expected to be available for such purposes during the period of interest. Various levels of improvement, some of which are listed below, may be undertaken to cope with the increasing demand pressures.

i) Augmentation of capacity in existing links. This improvement may be realizable by various means, ranging from enforced parking restrictions in certain arteries to new lane construction and more expedient traffic control systems.

ii) Rearrangement of one-way and two-way streets to provide an optimal configuration.

iii) Addition of new links to the existing network.

iv) For public transportation, construction of new terminal facilities and links to connect these facilities with already existing ones.

In practice, the final long-range transportation plan may call for a mixed strategy utilizing various modes of network improvement. It is obvious that a transportation planner has to analyze a large number of improvement alternatives, before a final plan is adopted. The two classes of traffic assignment models studied previously provide totally different approaches to solving such synthesis problems.

6.6 NETWORK SYNTHESIS VIA DESCRIPTIVE MODELS

To describe the synthesis solution when descriptive models are employed, let us assume that a specific set of links proposed for construction constitute the type of improvement prescribed.

It is obvious that each project may not be analyzed independently of the others, since the total network performance is highly dependent on the combination of projects considered. On the other hand, if m is the number of possible link additions, 2^m different alternatives exist, and its exhaustive analysis is clearly impossible for even moderately large m. The usual practice in the economic evaluation of traffic networks is to select *a priori*, a small subset of the potentially large number of alternative networks and accept the one that provides the best "measure of effectiveness".

To determine that measure of effectiveness, a traffic assignment is required for each alternative network as provided by a given descriptive model. The output of the traffic assignment (average daily traffic for each link) may be converted into users' cost. The accumulated users' cost for the entire network, and the total capital investment for the plan presently considered, are the parameters needed for estimating a measure of effectiveness for that project. A detailed description of the various elements required in such a process, as well as procedures and methods to obtain them, may be found in the work of Haikalis and Joseph [11]. When the process just described has been completed for the subset of plans under analysis, the usual practice is to apply a benefit-cost ratio analysis, a rate of return on marginal investment analysis, or some other classicial economic method, to determine the best alternative among those which have been preselected for consideration.

When a budgetary constraint is imposed on improvement expenditures, the synthesis problem may be solved by direct application of optimization methods for combinatorial problems. In particular, a direct search technique or an implicit enumeration method may be in order. We conjecture here that it may be desirable to apply an implicit enumeration technique as described below. First, we assume that as the number of links added to the network increases, the total user cost declines. Let the MOE be the user cost, with B the budgetary ceiling on capital investment, and $\underline{y}$ an m component binary vector associated with the acceptance or rejection of the links considered for construction. Hence, the new link addition problem may be expressed in terms of B and the project costs a_j as

$$S : \text{Minimize} \quad z = f(\underline{y}) \tag{6.1}$$

$$\text{Subject to} \quad \sum_{j=1}^{m} a_j y_j \le B \tag{6.2}$$

$$y_j = 0 \text{ or } 1 \tag{6.3}$$

where z is the total users cost as a function of the vector $\underline{y}$.

Problem S is a constrained optimization problem that may be interpreted as capital rationing for dependent projects. The dependency appears in the objective function (6.1), which cannot be expressed in closed mathematical form, but can only be evaluated as a result of a traffic assignment for each vector $\underline{y}$, (each network configuration) considered.

SOLUTION METHOD. We shall propose an implicit enumeration technique based on the general algorithm of Chapter II,(See[18] for a complete presentation), where

$$S_1 = \{y_j \ / \ 0 \leq y_j \leq 1\} \tag{6.4}$$

$$T_1 = \{y_j \ / \ \sum_{j=1}^{m} \beta_j y_j \leq B, \ y_j = 0 \text{ or } 1\} \tag{6.5}$$

$$\Omega_1 = S_1 \cap T_1 = T_1 \tag{6.6}$$

The auxiliary problem becomes:

$$A_\ell : \text{Minimize} \quad z = f(\underline{y})$$

$$\text{Subject to} \quad 0 \leq y_j \leq 1 \quad , j \in J$$

$$y_j = 0 \quad , j \in J_0$$

$$y_j = 1 \quad , j \in J_1$$

where J is the set of free links, J_0 the set of rejected links and J_1 the set of accepted links. Under the assumption that the value of z does not increase, as the number of links added to the network increases, the optimal solution to the optimization problem A_ℓ is

$$y_j^* = \begin{cases} 0, \text{ if } j \in J_0 \\ 1, \text{ otherwise} \end{cases} \tag{6.7}$$

where the value $z^*(\ell)$ is determined after the output of a computer program, which performs the descriptive traffic assignment and converts the link traffic volumes into user cost, is obtained. The branch and bound algorithm may then be stated as follows:

STEP 1. Set $i = 1$. Generate node 1 by solving a traffic assignment with $y_j^* = 1, \forall j$. Let $z^*(1)$ be the total user cost. Calculate $B^* = \sum_{j=1}^{m} a_j y_j^*$. If $B^* \leq B$, stop; the network configuration is optimal. Otherwise, bound node 1 with $L_1 = z^*(1)$. Set $i = i + 1$ and go to step i.

STEP i

a) BRANCH. Branch from bounded node ℓ. Select one y_k to be fixed to zero and one. Create nodes r and $r + 1$ and directed arcs (ℓ, r) and $(\ell, r + 1)$. Solve the traffic assignment corresponding to A_r with $y_k = 0$, adding k to the set J_0. Solve the traffic assignment corresponding to A_{r+1}, with $y_k = 1$, adding k to J_1.

b) BOUND. Select node ℓ such that $z^*(\ell) = \min_r \{z^*(r)\}$, for current terminal nodes. If $\sum_{j=1}^{m} a_j y_j^* \leq B$ for node ℓ, stop; the solution associated with node ℓ is optimal. Otherwise, set $i = i + 1$ and go to step i.

6.7 NETWORK SYNTHESIS VIA NORMATIVE MODELS

The important advantage of normative models lies in their flexible handling of synthesis problems, since the intrinsic nature of optimization problems is such that a convenient solution technique takes care of the combinatorial aspects, and finally selects the best project combination.

A substantial amount of research has been undertaken in this area, and various model formulations have evolved from the study of various types of network improvement problems.

The technique of continuous augmentation of capacity on existing links has been formulated by Garrison and Marble [12] and by Quandt [13]. In the latter model, the construction cost appears as a budgetary constraint, rather than as part of the objective function, as treated by Garrison and Marble.

Hershdorfer [9] studied the optimal one-way and two-way street configuration by extending Charnes and Cooper's multi-copy network model by an *ad hoc* introduction of decision variables into the model.

Hershdorfer, and also Roberts and Funk [14] have used Dantzig's scheme of introducing decision variables in the upper-bounding constraints on certain links, thereby obtaining a suitable formulation for the *new link addition* problem. Roberts and Funk consider rural network improvement subject to a budgetary constraint as opposed to Hershdorfer, who essentially assumes an infinite budget and congested networks. Recently, Ridley [15] has developed a combinatorial approach which he calls the "method of bounded subsets" for solution of the discrete augmentation of capacity problem.

The branch and bound algorithm developed in Section 6.6 is equally applicable to the new link addition synthesis problem when normative models are employed. In this case, the subprograms A_ℓ correspond to multi-copy network flow problems, which can be solved by means of a decomposition form linear programming code.

The simultaneous optimal node and link selection for an urban public transportation network, subject to a budgetary constraint, has been solved by Ichbiah [16] by means of a parametric branch and bound technique. His model does not directly consider flow volumes on the proposed network.

The set of models described above study network improvement problems for a single time period (base year to target year). In fact, the budget available for transportation investments is commonly appropriated in a multi-stage manner. Although the models indicated may be applied successively for various time increments, what the long-range transportation plan calls for is a <u>sequence</u> of improvements of the traffic network so that a convenient figure of merit is optimized over the total sequence of planning periods. The purpose of this chapter is to formulate a normative model that represents the goals indicated above, for different types of improvements.

6.8 NETWORK IMPROVEMENTS OVER TIME

Based on the discussion of previous sections, we may conclude that the preparation of a long-range transportation plan for a target year has been sufficiently studied, and that a variety of mathematical models are available to conveniently attack the problem. Considering the fact that a master plan would be actually implemented in a stage by stage fashion, and that available funds for transport expenditures are usually appropriated in fixed amounts for each planning period, a model taking this staging into account seems more appropriate.

Research on network improvement overtime has been done by Kalaba [16] for communication networks. The problem that he considers, however, is continuous augmentation of existing links' capacities. Roberts [17] has studied the multistaged link addition problem and proposes a solution method based on solving each stage, commencing from the last one, with a budget equal to the sum of the budgets up to the stage being considered. Links not accepted in the last period, are deleted from further consideration. His solution does not necessarily provide an optimum when the goal is to minimize a figure of merit over the entire horizon.

Before developing a normative model for the multistage link addition problem, we shall mention certain important aspects of the problem.

In the preparation of a transportation plan, before decisions can be made regarding facility improvements which are feasible in terms of cash flow, a preliminary planning of new facilities is required. The study of deficiencies in capacity provides a basis for such preliminary design. We shall assume that a set of possible new facilities, from which no optimum plan or subset is to be selected has been established.

We assume further that the construction costs for a specific type of facility have been previously obtained. This is obviously difficult since in order to obtain them, the facility must be located; and to estimate cost, certain standards must be fixed, which depend in general on the flow volumes likely to use the facility.

Finally, the future demands for transportation required by the model have been derived from forecasted land use patterns, but the new facilities provided in the planning period will in turn modify the land use development pattern. We shall not consider this interaction directly, but it could be treated with an iterative application of the model.

With these assumptions made we proceed to develop an optimization model that permits us to determine, for normative traffic behavior, the best improvement plan in time and space.

6.9 ANALYTICAL FORMULATION

Consider the graph $G = [\bar{N}, A]$ consisting of nodes which are denoted in any order by the sequence of numbers $1, 2, \ldots, N$ and of directed arcs (i, j) joining nodes of the set $\bar{N}$. The set of all arcs is denoted by A, and will be partitioned into two sets X and Y such that $X \cup Y = A$. The set X contains the set of all arcs of the existing network of interest. The set Y contains all arcs which form the set of proposals to be added to the network over n time periods. Note that if $X = \Phi$ we are confronted with a complete synthesis problem.

Let the amount of flow of copy α (here a *copy* associates all of the traffic flowing from or to a specific origin or destination) associated with arc $(i, j) \varepsilon A$ at stage k be x^{α}_{ijk}. Denote by c_{ijk} the discounted unit cost of travel on arc (i, j) at stage k, and by u_{ijk} the capacity or upperbound on the flow over arc (i, j) at stage k.

For each $(i, j) \varepsilon Y$, let a_{ijk} be the capital outlay required to build arc (i, j) if selected for construction at period k.

Denote by $r^{\alpha}_{ik} > 0$ the net amount of flow into node i of copy α at time period k, (or $r^{\alpha}_{ik} < 0$ if the net flow is out), and by E, the node-arc incidence matrix which describes the network G. The total budget ceilings available at time period k will be denoted by B_k. Let n be the number of time periods and N the total number of copies.

The problem of optimally selecting link proposals for construction is that of satisfying the budgetary constraints at each period and minimizing the total user cost over the entire interval. It may be formulated as follows:

$$M : \text{Minimize} \quad z = \sum_{\alpha=1}^{N} \sum_{k=1}^{n} \sum_{(i,j)\varepsilon A} c_{ijk}\, x_{ijk}^{\alpha} \tag{6.8}$$

$$\text{Subject to} \quad \sum_{j=1}^{N} e_{ij}\, x_{ijk}^{\alpha} = r_{ik}^{\alpha} \quad , \forall\, i, \alpha, k \tag{6.9}$$

$$\sum_{\alpha=1}^{N} x_{ijk}^{\alpha} \leq u_{ijk} \quad , (i,j)\varepsilon X, \forall\, k \tag{6.10}$$

$$\sum_{\alpha=1}^{N} x_{ijk}^{\alpha} \leq u_{ijk}\, y_{ijk} \, , (i,j)\varepsilon Y, \forall\, k \tag{6.11}$$

$$y_{ijk} - y_{ijk+1} \leq 0 \quad , (i,j)\varepsilon Y, k=1, \ldots, n-1, n > 1 \ (*) \tag{6.12}$$

$$\sum_{(i,j)\varepsilon Y} a_{ijk}\,(y_{ijk} - y_{ijk-1}) \leq B_k, \forall\, k \tag{6.13}$$

$$x_{ijk}^{\alpha} \geq 0 \tag{6.14}$$

$$y_{ijk} = 0 \text{ or } 1 \tag{6.15}$$

In this formulation, constraints (6.9) represent the conservation of flow equations for all copies and all time periods, with e_{ij} being the corresponding element of the node arc incidence matrix E.

The constraints (6.10) constitute the upperbounding constraints on the sum of all copy-flows utilizing originally existent arcs $(i,j)\ \varepsilon\ X$. For proposed arcs, $(i,j)\ \varepsilon\ Y$, a set of decision variables y_{ijk} has been introduced which can take the binary values 1 or 0 as indicated by (6.15), depending on whether or not arc $(i,j)\ \varepsilon\ Y$ is available for use at time

---

* For n=1, constraints (6.12) have no particular meaning and may be dropped from further consideration.

period k. Constraints (6.11) therefore guarantee that for a non-constructed arc, the corresponding flows will vanish ($y_{ijk} = 0$), or otherwise that they do not exceed the provided capacity ($y_{ijk} = 1$)

The set (6.12) acts as a "turn-on switch", guaranteeing that if a certain link is adopted for investment at time period s (i.e., $y_{ijs} = 1$, with $y_{ijk} = 0$, $k < s$), it will remain available for utilization in subsequent periods (i.e., $y_{ijk} = 1$, $k > s$).

Constraints (6.13) represent the budgetary constraints; here $y_{ijo} \equiv 0$ for all $(i,j) \in Y$. The difference of the decision variables for two subsequent time periods, in addition to (6.12), guarantee that a single capital outlay is disbursed for each project.

If the problem being considered calls for multiple outlays once a link has been selected for construction, e.g. when maintenance costs are considered, it suffices to modify the budgetary constraints (6.13) and replace it by

$$\sum_{(i,j)\in Y} a_{ijk}\, y_{ijk} \le B_k \quad , \forall k \tag{6.16}$$

Finally, relation (6.14) simply expresses the non-negativity conditions on the arc flows for all time periods.

REMARKS. We observe that for the single time period case, the index k may be dropped; constraints (6.12) will no longer have any meaning and may also be dropped. Constraints (6.13) reduce to a single budgetary constraint and problem M becomes the link-addition problem as formulated by Roberts [17], except for the fact that construction costs are not part of the objective.

If both indices α and k are relaxed, the resulting model becomes a single period problem of capital investment in links of a general homogeneous commodity network.

This problem is somewhat similar to the knapsack problem considered in Chapter IV. The main difference, however, is that the payoff function for a certain combination of links may not be determined until a cost minimization network flow problem for the configuration under analysis is solved.

Finally we remark that the objective function is linear and represents the total user cost over the entire interval of interest. Therefore, model M in its most general form is a (0-1) mixed-integer linear program of a very complex nature, as may be immediately recognized, but with important structural characteristics that we shall identify in the following section.

6.10 STRUCTURAL CHARACTERISTICS OF THE MODEL

Let us assume that the node-arc incidence matrix E describing the network $G = [\bar{N}, A]$ is constructed in such a way that all arcs $(i,j) \in X$ occupy the first part of the matrix, while arcs $(i,j) \in Y$ will be associated with the remaining columns of E. Accordingly, we define the following partition for E, $E = [\bar{E}, \hat{E}]$.

Let $\underline{\bar{x}}_k^\alpha$ and $\underline{\hat{x}}_k^\alpha$ be the flow vectors for arcs $(i,j) \in X$ and $(i,j) \in Y$ respectively, for copy α in time period k. Denote by $\underline{y}_k$ the vector of all decision variables at time period k, by $\underline{r}_k^\alpha$ the demand vector for copy α in period k, and by $\underline{u}_k$ the upper bound vector on the flow of arcs $(i,j) \in X$ at time period k. Finally, according to the partition defined for E, let $\underline{\bar{c}}_k^\alpha$ and $\underline{\hat{c}}_k^\alpha$ be the user cost vectors for arcs $(i,j) \in X$ and $(i,j) \in Y$ respectively, for copy α in time period k.

Our model M may then be rewritten in the condensed form depicted in Table 6-1. Here I is the identity matrix, U_k is a diagonal matrix having the upperbounds u_{ijk} for $(i,j) \in Y$ as diagonal elements, and $\underline{a}_k$ is the vector representing capital outlays in period k for all $(i,j) \in Y$. The non-negativity conditions on the flows and the binary values of the y_{ijk}, although not explicitly indicated in Table 6-1, are to be satisfied.

The arrangement of the variables in Table 6-1 is highly suggestive of a partition into two sets, the first embodying the decision variables $\underline{y}_k$, $\forall$ k, and the second all the flow variables for all time periods. Furthermore, Table 6-1 presents a similar structure to that of the class of problems presented in Appendix A, (see Page A-3), with additional simplifications. Indeed, using the notation of the Appendix, we observe that all the B matrices are identical in our problem and are highly structured as well, suggesting that additional exploitation is possible.

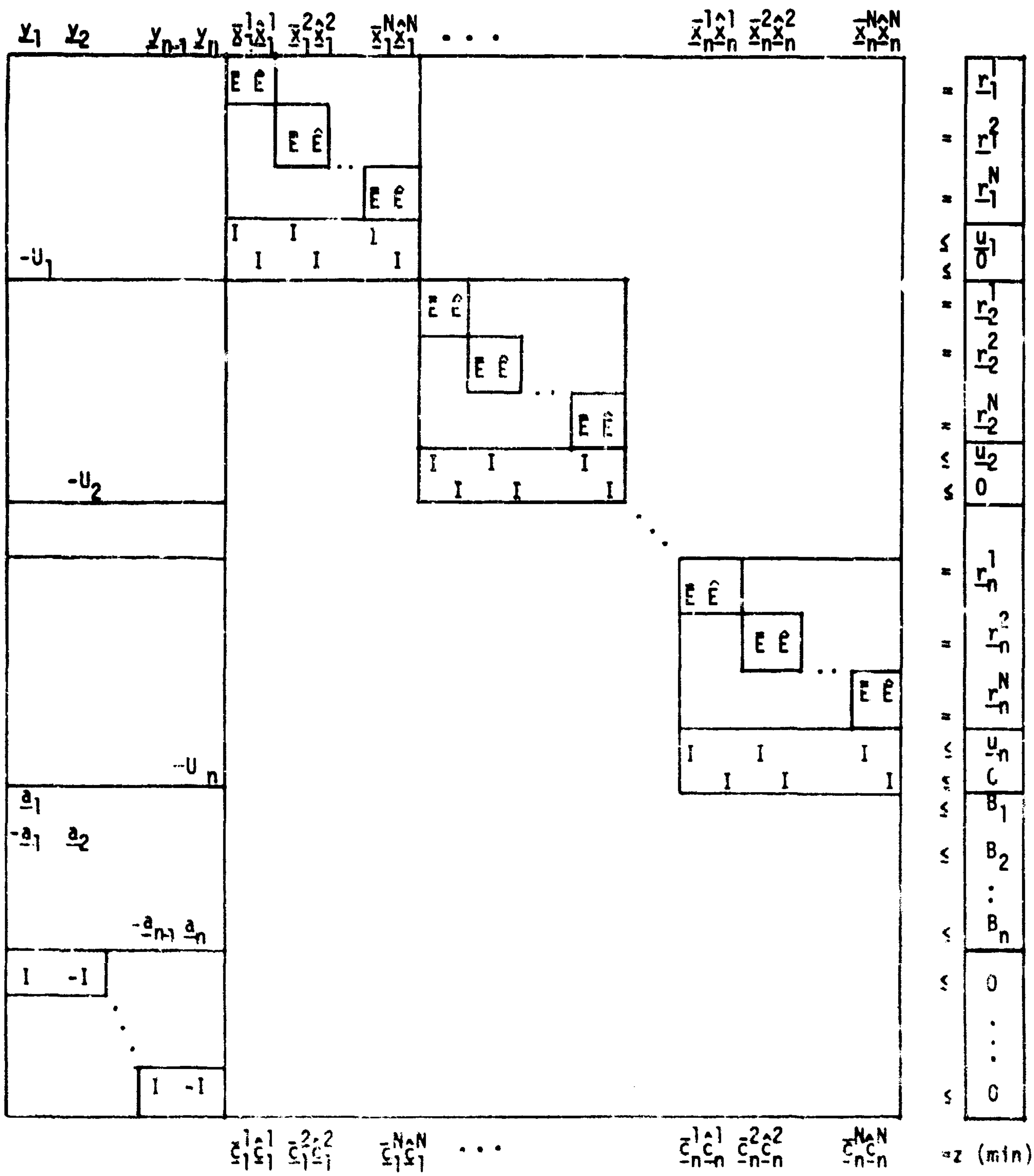


TABLE 6-1

The A matrices are composed mostly of zeros except for the diagonal submatrices U_k. Finally, observe that $\underline{c}_0$, the cost associated with the binary variables, is zero in this case, a fact which will further simplify calculations.

6.11 SOLUTION METHOD BY PARTITIONING

In this section we discuss the partitioning technique of Benders as presented in Appendix A when it is applied to our multistaged link addition problem M. We shall use approximately the notation of Appendix A. In the present case, the set S_0 is defined by all y_{ijk} satisfying constraints (6.12), (6.13), and (6.15). At each step of the algorithm, and once the auxiliary (0-1) problem B' has been solved yielding the optimal values $\tilde{y}^0_{ijk}$, these will be used to solve each one of the subprograms:

$$P_k : \text{Minimize} \quad z = \sum_{\alpha=1}^{N} \sum_{(i,j)\varepsilon A} c_{ijk}\, x^{\alpha}_{ijk} \tag{6.17}$$

$$\text{Subject to} \quad \sum_{j=1}^{N} e_{ij}\, x^{\alpha}_{ijk} = r^{\alpha}_{ik},\ \forall\ i,\ \alpha \tag{6.18}$$

$$\sum_{\alpha=1}^{N} x^{\alpha}_{ijk} \le u_{ijk} \quad , (i,j)\ \varepsilon\ X \tag{6.19}$$

$$\sum_{\alpha=1}^{N} x^{\alpha}_{ijk} \le u_{ijk}\, \tilde{y}^{0}_{ijk} \quad , (i,j)\ \varepsilon\ Y \tag{6.20}$$

$$x^{\alpha}_{ijk} \ge 0,\ (i,j)\ \varepsilon\ A,\ \forall\ \alpha \tag{6.21}$$

for all values of k (all time periods). However, problem P_k is essentially a multi-copy network flow problem and therefore, at each iteration of the partitioning algorithm, n problems of the form P_k must be solved for the given values $\tilde{y}^0_{ijk}$. Each of these problems presents in turn a block-angular structure and thus a higher level of decomposition may be applied.

For example, the Dantzig and Wolfe decomposition method, as investigated by Pinnell [8] may be applied. When this scheme is used, the solution of each subproblem at each iteration of the decomposition procedure reduces to finding its shortest path tree.

As for the second part of the partitioning algorithm, the solution of an all-integer (0-1) programming problem is required with one or two additional constraints at each iteration.

The problem presents the following form:

$$B' : \text{Minimize} \quad z = \sum_{k=1}^{n} y_k \tag{6.22}$$

Subject to

$$\sum_{i=1}^{N} \hat{\pi}_{ikt}\, r_{ik}^{\alpha} - \sum_{(i,j)\varepsilon X} \hat{\pi}_{ijkt}\, u_{ijk} - \sum_{(i,j)\varepsilon Y} \hat{\pi}_{ijkt}\, u_{ijk}\, y_{ijk} \le y_k, \quad t \ \varepsilon\ T_k \tag{6.23}$$

$$y_{ijk} - y_{ijk+1} \le 0 \quad , (i,j)\ \varepsilon Y\ , \ k=1,\ldots, n-1 \tag{5.24}$$

$$\sum_{(i,j)\varepsilon Y} a_{ijk}\, (y_{ijk} - y_{ijk-1}) \le B_k, \ \forall\ k \tag{6.25}$$

$$y_{ijk} = 0 \text{ or } 1 \quad , \forall\ (i,j)\ \varepsilon\ Y, \ \forall\ k \tag{6.26}$$

$$y_k \text{ unrestricted}, \ \forall\ k \tag{6.27}$$

where $[\hat{\pi}_{ikt}, \hat{\pi}_{ijkt}]$ are the components of an extreme point $t\ \varepsilon\ T_k$ of the polytope associated with the dual of P_k.

For the one copy case ($n = 1$) and for integer demands and capacities, network flow theory shows that, the dual variables $\underline{\pi}$ are also integers. Hence, from (6.23), the y_k are integers. Problem B' is therefore an all-integer program from which a feasible solution is already available (namely, $y_{ijk} = 0$, $(i,j)\ \varepsilon\ Y$, $\forall\ k$) and the Young-González algorithm may be applied. For the multi-copy case, however, the $\underline{\pi}$ values are not necessarily integer and the y_k may not be integer. B', in this case, is a mixed-integer linear program.

We conclude with the following remark on problems with a more flexible budget structure.

REMARK. The basic model considered in formulation M may be extended for the case of budget ceiling deferrals. The mechanism that permits funds which remain unused in a certain period, to be transferred to a later period is given by a proper manipulation of the slack variables of constraints (6.13).

Let s_k, $k = 1, \ldots, n$ be the amount of unused funds of period k. The constraints (6.13) take the new form

$$\sum_{(i,j)\in Y} a_{ijk} \left(y_{ijk} - y_{ijk-1}\right) - s_{k-1} + s_k = B_k, \; k=1,\ldots,n \qquad (6.28)$$

with $s_0 \equiv 0$, and with s_n representing the idle funds, if any, at the end of the assignment. We observe that no present worth factor is attached to the variables s_k, so they represent for all cases, simply idle cash.

The solution to M with (6.28) instead of (6.13) is not substantially altered. Its influence will be reflected exclusively in the solution of the integer program, B', by augmenting the problem with the n slack variables s_n. If a_{ijk} and B_k are assumed to be integer, then s_n, $\forall_n$ will also be integer.

6.12 NOTES TO CHAPTER VI

[1] Moyer, R. A., "Comprehensive Urban Transportation Study Methods", A.S.C.E., *Journal of the Highway Division*, December 1965, pp. 57-87.

[2] Martin, B. V., F. W. Memmott, III, and A. J. Bone, *Principles and Techniques of Predicting Future Demand for Urban Area Transportation*, The M.I.T. Pr ss, June 1961.

[3] Wardrop, J. G., "Some Theoretical Asp ts of Road Traffic Research", *Inst. of Civ. Eng.*, Road Paper No. 36, Lond, 1952.

[4] Beckman, M., C. B. McGuire, and C. Winster, *Studies in the Economics of Transportation*, New Haven, Connecticut: Yale University Press, 1956.

[5] Charnes, A. and W. W. Cooper, "Multi-Copy Traffic Network Models", in *Proc. Symposium on the Theory of Traffic Flow*, R. Herman (ed.), New York: Elsevier, 1961, pp. 85-96.

[6] Bone, A. J., *Final Report of the Highway Transportation Demand Research Project*, Research Report R65-24, M.I.T., Department of Civil Engineering, No. 1965.

[7] Martin, B. V., *A Computer Program for Traffic Assignment Research*, Research Report R64-41, M.I.T., Department of Civil Eng...ing, December 1964.

[8] Pinnell, C. and G. T. Satterly, Jr., "Analytical Methods in Transportation: Systems Analysis for Arterial Street Operation", A.S.C.E., *Journal of the Engineering Mechanics Division*, December 196[illegible], pp. 63-95.

[9] Hershdorfer, A. M., *Optimal Routine of Urban Traffic*, Ph.D. Thesis, Massachusetts Institute of Technology, June 1965.

[10] Jorgensen, N. O., *Some Aspects of the Urban Traffic Assignment Problem*, Graduate Report, Institute of Transportation and Traffic Engineering, University of California, June 1963.

[11] Haikalis, G. and H. Joseph, "Economic Evaluation of Traffic Networks", *H. R. B. Bulletin 306*, 1961, pp. 39-63.

[12] Garrison, W. L. and D. F. Marble, "Analysis of Highway Networks: A Linear Programming Formulation", *H. R. B. Proceedings, Vol.* 37, 1958, pp. 1-17.

[13] Quandt, R. E., "Models of Transportation and Optimal Network Construction", *Journal of Regional Science*, Vol. 2, 1960, pp. 27-45.

[14] Roberts, P. O., and M. L. Funk, *Toward Optimum Methods of Link Addition in Transportation Networks*, Massachusetts Institute of Technology, Department of Civil Engineering, September 1964.

[15] Ridley, T. M., *An Investment Policy to Reduce the Travel Time in a Transportation Network*, Operations Research Center, ORC 65-34, University of California, Berkeley, December 1965.

[16] Kalaba, R. E., and M. L. Juncosa, "Optimal Design and Utilization of Communication Networks", *Management Science*, Vol. 3, 1956, pp. 33-44.

[17] Roberts, P. O., *Transport Planning: Models for Developing Countries*, Ph.D. Thesis, Department of Civil Engineering, Northwestern University, 1966.

[18] Ochoa-Rosso, F., and A. Silva, *Optimum Project Addition in Urban Transportation Networks via Descriptive Traffic Assignment Models* , Research Report R68-41, Massachusetts Institute of Technology, Department of Civil Engineering, June, 1968.

APPENDIX A

THE DECOMPOSITION PRINCIPLE FOR MIXED-VARIABLE PROGRAMS

A.1 INTRODUCTION

In this Appendix we shall present in detail the partitioning technique developed by Benders [1] as applied to both continuous and mixed-integer linear programming problems presenting the so-called *block-angular* structure. In the case of continuous linear programs the method, as noted by Benders and also by Balinski [2], constitutes a dual form of the Dantzig and Wolfe decomposition principle [3]. We shall explicitly show this property.

In the linear programming case, the partitioning technique requires the solution of a linear program differing from the one of the previous iteration by one or two constraints. Thus the *dual-simplex* method is indicated to reoptimize the problem subject to the additional constraint(s).

For the case of the (0-1) mixed-integer linear program, the partitioning technique requires the solution of an all-integer (0-1) programming problem augmented at each iteration by one or two additional constraints. The Young-González algorithm [4], [5] is the method we have selected for the solution of the all-integer program, and a procedure to reoptimize the solution of the previous iteration is indicated.

The partitioning technique developed here is directly applied in chapters V and VI to the solution of multi-stage network synthesis problems.

A.2 PROBLEM FORMULATION. DERIVATION OF AN EQUIVALENT PROGRAM

We shall consider the class of mathematical programming problems with the following analytical formulation:

A : Determine $\underline{x}_0^0$ and $\underline{x}_i^0$ so as to

$$\text{Minimize} \quad z = \underline{c}_0 \underline{x}_0 + \sum_{i=1}^{m} \underline{c}_i \underline{x}_i \tag{A.1}$$

$$\text{Subject to} \quad A_0 \underline{x}_0 = \underline{b}_0 \tag{A.2}$$

$$A_i \underline{x}_0 + B_i \underline{x}_i = \underline{b}_i \ , \ i=1, \ldots, m \tag{A.3}$$

$$\underline{x}_i \geq 0 \ , \ i=1, \ldots, m \tag{A.4}$$

$$\underline{x}_0 \in \bar{S}_0 \tag{A.5}$$

where A_0 is an $(m_0 \times n_0)$ matrix, A_i is an $(m_i \times n_0)$ matrix, B_i an $(m_i \times n_i)$ matrix, $\underline{x}_i$ and $\underline{c}_i$ are vectors with n_i components and $\underline{b}_i$ are vectors with m_i components. The n_0 component vector $\underline{x}_0$ is defined over the region $\bar{S}_0$ We shall define S_0 as the intersection of (A.2) and (A.5) for the following special cases of $\bar{S}_0$:

i) $\bar{S}_0$ the non-negative orthant. Therefore

$$S_0 = \{\underline{x}_0 \ / \ A_0 \underline{x}_0 = \underline{b}_0 \ , \ \underline{x}_0 \geq 0\} \tag{A.6}$$

ii) $\bar{S}_0$ the discrete set defined by the vertices of the unit hypercube. Therefore

$$S_0 = \{\underline{x}_0 \ / \ A_0 \underline{x}_0 = \underline{b}_0, \ x_{0j} = 0 \text{ or } 1, \ j=1, \ldots, n_0\} \tag{A.7}$$

For case i), problem A becomes a linear program in standard form that may be solved by applying the decomposition principle of Dantzig and Wolfe to its dual program. For case ii), the resulting program is a (0-1) mixed integer programming problem. If each $A_i = 0$, $i = 0,1,\ldots, m$ the problem reduces to a set of *mutually independent* problems of the form Min $z_i = \underline{c}_i \underline{x}_i$, $B_i \underline{x}_i = \underline{b}_i$, $\underline{x}_i \geq 0$ In any case, the constraints of problem A present the following associated structure of block-angular form:

$$\begin{bmatrix} A_0 & & & & \\ A_1 & B_1 & & & \\ A_2 & & B_2 & & \\ \vdots & & & \ddots & \\ A_m & & & & B_m \end{bmatrix} \begin{bmatrix} \underline{x}_0 \\ \underline{x}_1 \\ \underline{x}_2 \\ \vdots \\ \underline{x}_m \end{bmatrix} = \begin{bmatrix} \underline{b}_0 \\ \underline{b}_1 \\ \underline{b}_2 \\ \vdots \\ \underline{b}_m \end{bmatrix} \qquad (A.8)$$

We shall now develop a program equivalent to problem A. Denoting by $\underline{x} = [\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_m]$ a solution vector to A, the problem may be expressed in the equivalent form

$$\underset{\underline{x}_0 \in S_0}{\text{Minimize}} \left[\underline{c}_0 \underline{x}_0 + \min_{\underline{x}} \left\{ \sum_{i=1}^{m} \underline{c}_i \underline{x}_i \;/\; B_i \underline{x}_i = \underline{b}_i - A_i \underline{x}_0, \underline{x}_i \geq 0 \right\} \right] \qquad (A.9)$$

The minimization problem within *curly* brackets, for a given value of $\underline{x}_0 \in S_0$, becomes a standard linear program in $\underline{x}$, which we denote as P. Note that solving P is equivalent to solving the following set of m mutually independent linear programs and summing up their objective function values:

P_i : Minimize $Z_i = \underline{c}_i \underline{x}_i$

Subject to $B_i \underline{x}_i = \underline{b}_i - A_i \underline{x}_0$; $i = 1, \ldots, m$

$\underline{x}_i \geq 0$

Problem P has an associated dual program D that may be decomposed into the following set of subprograms corresponding to the dual programs of P_i:

$$D_i : \text{Maximize} \quad \bar{z}_i = (\underline{b}_i - A_i \underline{x}_0)' \underline{u}_i$$

$$\text{Subject to} \quad B_i' \underline{u}_i \leq \underline{c}_i'$$

$$\underline{u}_i \quad \text{unrestricted}$$

where the symbol (') indicates transpose. By the duality theorem of linear programming, if feasible solutions exist for both P_i and D_i, then their optimal solutions z_i^0 and $\bar{z}_i^0$ satisfy $z_i^0 = \bar{z}_i^0$.

Hence, expression (A.9) may be expressed in terms of the dual problems D_i as follows:

$$\underset{\underline{x}_0 \in S_0}{\text{Minimize}} \left[\underline{c}_0 \underline{x}_0 + \sum_{i=1}^{m} \max_{\underline{u}_i} \{(\underline{b}_i - A_i \underline{x}_0)' \underline{u}_i \ / \ B_i' \underline{u}_i \leq \underline{c}_i'\} \right] \quad (A.10)$$

Consider the convex polytope (a finite number of closed halfspaces) $S_i = \{\underline{u}_i \ / \ B_i' \underline{u}_i \leq \underline{c}_i'\}$. Observe that S_i is independent of the values of $\underline{x}_0$. We shall assume temporarily that S_i is bounded (i.e., it is a convex polyhedron). Then from (A.10) note that for *any value* of $\underline{x}_0 \in S_0$, the maximum of each subprogram D_i will occur at an extreme point of S_i. Denote by $\hat{\underline{u}}_{ik}$, $k \in K_i$ the extreme points of the polyhedron $B_i' \underline{u}_i \leq \underline{c}_i'$. We shall assume that there are N_i such extreme points. Hence it suffices to calculate the values of $(\underline{b}_i - A_i \underline{x}_0)' \hat{\underline{u}}_{ik}$ for each extreme point and select the maximum value, yielding the solution to D_i for a given value of $\underline{x}_0$.

From the above discussion, (A.10) is equivalent to

$$\underset{\underline{x}_0 \in S_0}{\text{Minimize}} \left[\underline{c}_0 \underline{x}_0 + \sum_{i=1}^{m} \max_{k \in K_i} \{(\underline{b}_i - A_i \underline{x}_0)' \hat{\underline{u}}_{ik}\} \right] \quad (A.11)$$

Let $y_i = \max_{k \in K_i} \{(\underline{b}_i - A_i \underline{x}_0)' \hat{\underline{u}}_{ij}\}$; then for each extreme point $\hat{u}_{ik}$ the following condition holds:

$$(\underline{b}_i - A_i \underline{x}_0)' \hat{\underline{u}}_{ik} \leq y_i \quad (A.12)$$

If we proceed analogously for all extreme points of each S_i, then (A.11) may be explicitly written in the form:

$$\text{B : Minimize} \quad z = \underline{c}_0\, \underline{x}_0 + \sum_{i=1}^{m} y_i \tag{A.13}$$

$$\text{Subject to } (\underline{b}_i - A_i\, \underline{x}_0)'\, \hat{\underline{u}}_{ik} \leq y_i \ , \ i=1,\ldots,m,\ k\epsilon K_i \tag{A.14}$$

$$\underline{x}_0 \ \epsilon\ S_0 \tag{A.15}$$

$$y_i \ \text{unrestricted} \quad , \ i=1,\ldots,m \tag{A.16}$$

The relationships (A.13) to (A.16) define a new program in terms of the variables $\underline{x}_0$ and y_i, which we shall now show to be *equivalent* to problem A.

If $(\underline{x}_0^0, \underline{x}_i^0)$ is an optimal solution to A, then $\underline{x}_i^0$ is an optimal solution to P_i for $\underline{x}_0 = \underline{x}_0^0$. Also, D_i will have an optimal solution for a certain extreme point $\hat{\underline{u}}_{ik}^0$. Hence, in problem B, expressions (A.14) will be satisfied, and those corresponding to $\hat{\underline{u}}_{ik}^0$ will be satisfied as strict equalities, thus $y_i^0 = (\underline{b}_i - A_i\, \underline{x}_0^0)'\, \hat{\underline{u}}_{ik}^0 = \underline{c}_i\, \underline{x}_i^0$. This implies that the optimal solution to B, $(\underline{x}_0^0, y_i^0)$, gives the same value for the objective function as that one obtained by A.

Conversely, assume that $(\underline{x}_0^0, y_i^0)$ is an optimal solution to B. Then for each i, at least one value of $\hat{u}_{ik}$ will satisfy (A.14) as a strict equality, say $\hat{\underline{u}}_{is}^0$ corresponding to the extreme point optimal solution of D_i. Then by solving the problems P_i for $\underline{x}_0 = \underline{x}_0^0$, we will obtain $\underline{x}_i^0$ with $\sum_{i=1}^{m} \underline{c}_i\, \underline{x}_i^0 = \sum_{i=1}^{m} y_i^0$. Since the optimal solution to A for a value $\underline{x}_0 = \underline{x}_0^0$ corresponds to the solution of P_i, it follows that $(\underline{x}_0^0, \underline{x}_i^0)$ is the optimal solution to A with the same value of the objective function as the one obtained for B.

For a formal proof considering the unbounded as well as the infeasible case the reader is referred to the work of Benders [1].

We shall focus our attention upon the solution of problem B for the two special sets S_0 defined at the beginning of the section.

A.3 THE CASE WHERE $\bar{S}_0$ IS THE NON-NEGATIVE ORTHANT

We consider problem B with S_0 given by the expression (A.6). Problem B, which we shall call the *master program*, becomes

$$B: \text{Minimize} \quad z = \underline{c}_0 \, \underline{x}_0 + \sum_{i=1}^{m} y_i \qquad , i=1,\ldots,m \qquad \text{(A.17)}$$

$$\text{Subject to} \quad (A_i' \, \hat{\underline{u}}_{ik})' \, \underline{x}_0 + y_i \geq \underline{b}_i' \, \hat{\underline{u}}_{ik}, \; k=1,\ldots,N_i \qquad \text{(A.18)}$$

$$A_0 \, \underline{x}_0 = \underline{b}_0$$

$$x_0 \geq 0 \quad , y_i \text{ unrestricted}$$

Thus, in passing from the linear program A in standard form to the equivalent *linear program* B, by partitioning the set of variables into $\underline{x}_0$ and the remaining $\underline{x}_i$ we have reduced the number of variables from $\sum_{i=0}^{m} n_i$ to $(n_0 + m)$. At the same time, we have increased the number of constraints from $\sum_{i=0}^{m} m_i$ to $(m_0 + \sum_{i=1}^{m} N_i)$.

Direct solution of problem B hardly produces a positive net result, since such an approach implies the calculation in advance of all constraints of type (A.18), i.e. the calculation of all the extreme points of the convex polyhedra S_i. Moreover, if an optimum solution to B is obtained, say $(\underline{x}_0^0, y_i^0)$, then the solution to each of the m linear programs P_i for $\underline{x}_0 = \underline{x}_0^0$ is required in order to find the corresponding optimal vectors $\underline{x}_i^0$ (i=1, ... ,m).

However, in the optimal solution to B only a subset of (A.18) will be active. The Benders' algorithm for solution of problem B (cf. Section A.5) makes use of this fact in attempting to generate those constraints (A.18) that determine optimality for B. The procedure solves B with a small subset of constraints (A.18); if optimality is not obtained, the *subprograms* D_i are solved to generate additional constraints to the *master program* B, which in turn will have to be reoptimized. This alternative process is repeated until an optimal solution (if one exists) is obtained in a finite number of steps, guaranteed by the fact that the number of constraints (A.18) is finite.

The solution procedure just outlined closely parallels the *decomposition algorithm* of Dantzig and Wolfe. We shall now interpret it in detail.

By obtaining the dual of problem B we have

$$C: \text{Maximize} \quad \bar{z} = \sum_{i=1}^{m} \sum_{k=1}^{N_i} f_{ik} \lambda_{ik} \qquad \text{(A.19)}$$

$$\text{Subject to} \quad \sum_{i=1}^{m} \sum_{k=1}^{N_i} \underline{d}_{ik} \lambda_{ik} + A_0 \underline{w}_0 \leq \underline{c}_0 \qquad \text{(A.20)}$$

$$\sum_{k=1}^{N_i} \lambda_{ik} = 1 \qquad \text{(A.21)}$$

$$\lambda_{ik} \geq 0$$

$$\underline{w}_0 \quad \text{unrestricted}$$

where $f_{ik} = \underline{b}_i' \, \hat{\underline{u}}_{ik}$ and $\underline{d}_{ik} = (A_i' \, \hat{\underline{u}}_{ik})'$

We observe that problem C corresponds to the so-called *master* or *extremal problem* of the *dual* of A, in the context of the Dantzig and Wolfe decomposition principle. This justifies the name that we have assigned to B in describing Benders' decomposition principle. In C, the variables λ_{ik}, $(i=1, \ldots, m;\ k=1, \ldots, N_i)$ are weights forming a convex combination of the extreme points $\hat{\underline{u}}_{ik}$ of the polyhedron S_i.

In the optimal solution to problem C, only a small subset of the variables λ_{ik} will be basic. The Dantzig and Wolfe method for solution of the dual program of A makes use of this fact; it tries to find the subset of λ_{ik} that determines optimality for C without examining all basic feasible solutions. This method first obtains a basic feasible solution for C; if the solution is not optimal, the subprograms D_i are solved to generate a new column (i.e., *proposal vector*) that should go into the basis of the master program C. This process is repeated until an optimal solution (if one exists) is obtained in a finite number of steps, guaranteed by the fact that the number of extreme points of the subprograms D_i is finite.

Finally we observe that for optimal solutions to B and C, (assuming non-degeneracy), to each basic variable λ_{1k}^{o} of C there corresponds an active constraint of (A.18). This follows from the complementary slackness conditions:

$$\lambda_{1k}^{o} > 0 \Rightarrow (A_1' \, \underline{\hat{u}}_{1k}) \, \underline{x}_o^{o} + y_1^{o} = \underline{b}_1 \, \underline{\hat{u}}_{1k}.$$

A.4 EXTENSION TO THE CASE WHERE THE S_1 ARE NOT ALL BOUNDED

If S_1 is an unbounded polytope, from convex set theory we know that it possesses a finite number of extreme points $\underline{\hat{u}}_{1k}$, $k \in K_1$ and a finite number of *extreme rays* $\underline{\bar{u}}_{1\ell}$, $\ell \in L_1$, emanating from certain extreme points.

Hence it may occur that for a certain value of $\underline{x}_0 \in S_0$ in (A.11), the solution to one of the dual programs D_1 tends to infinity (i.e., problem D_1 is unbounded) along the half line

$$\{\underline{u}_1 \,/\, \underline{u}_1 = \underline{\hat{u}}_{1k} + \delta \underline{\bar{u}}_{1\ell}, \; k \in K_1, \; \ell \in L_1, \; \delta \geq 0\}.$$

The corresponding value of the objective function may be expressed as

$$z_1 = (\underline{b}_1 - A_1 \, \underline{x}_0)' \, \underline{\hat{u}}_{1k} + \delta(\underline{b}_1 - A_1 \, \underline{x}_0)' \, \underline{\bar{u}}_{1\ell}$$

If $z_1 \to \infty$, and since $\delta \geq 0$ from the previous expression, it follows that

$$(\underline{b}_1 - A_1 \, \underline{x}_0)' \, \underline{\bar{u}}_{1\ell} > 0 \qquad \text{(A.22)}$$

If D_1 is unbounded, problem P_1 and thus problem A are infeasible for values of $\underline{x}_0 \in S_0$ for which (A.22) holds. Hence, to prevent $\underline{x}_0$ from taking on such values, it suffices to restrict (A.11) or its equivalent problem B with the following constraints associated with all extreme rays of S_1:

$$(\underline{b}_1 - A_1 \, \underline{x}_0)' \, \underline{\bar{u}}_{1\ell} \leq 0 \; ; \; \ell \in L_1 \; , \; \forall_1 \qquad \text{(A.23)}$$

The extreme rays of S_i may be obtained by finding the extreme rays of its associated *polyhedral convex cone* $\{\underline{u}_i \,/\, B_i' \underline{u}_i \leq 0\}$. In this manner the set L_i is determined and the set of constraints (A.23) may be constructed.

If constraints (A.23) thus constructed are added to problem B, then *any* feasible solution $(\underline{x}_0, y_i)$ to B, with $\underline{x}_0$ then used in solving the problems P_i, will result in a feasible solution $(\underline{x}_0, \underline{x}_i)$ to the original problem A.

Finally consider the linear programming case of section A.3 with constraints (A.23) included. Note that the resulting dual program C is analogous to the one obtained by applying the Dantzig and Wolfe method to the dual of a problem A containing unbounded subprograms.

A.5 THE PARTITIONING ALGORITHM OF BENDERS

The Benders' partitioning algorithm, instead of solving directly problem B, solves iteratively a less restricted problem B' with the same objective as B. At each new iteration, additional constraints are added to B' and the problem is reoptimized. Since eventually B' would be identical to B, the optimal solution to the latter must finally be found. However, the method tries to reach optimality for B by solving problems B' with only a small subset of the total number of constraints.

Let I be the set of indices $i=1,\ldots,m$, and I' a subset of I. Also let K_i' be a subset of the set of extreme points K_i to subprogram i, and $L_i' \subseteq L_i$. Then problem B' may be formulated as follows:

B' : Determine $\tilde{\underline{x}}_0^0$, $\tilde{y}_i^0$ and $\tilde{z}^0$ so as to

Minimize $\tilde{z} = \underline{c}_0\, \tilde{\underline{x}}_0 + \sum_{i=1}^{m} \tilde{y}_i$

Subject to $(\underline{b}_i - A_i\, \tilde{\underline{x}}_0)'\, \underline{u}_{ik} \leq \tilde{y}_i \;;\; i \in I',\ k \in K_i'$ (A.24)

$(\underline{b}_i - A_i\, \tilde{\underline{x}}_0)'\, \bar{\underline{u}}_{i\ell} \leq 0 \;;\; i \in I',\ \ell \in L_i'$ (A.25)

$\tilde{\underline{x}}_0 \in S_0$; $\tilde{y}_i$ unrestricted; $i=1,\ldots,m$

Assuming that an optimal solution with value z^0 exists for B, and that $\hat{z}^0$ is an optimal solution to B', it follows that $\hat{z}^0 \leq z^0$, since B' is *less* restricted than problem B. That is, the solution to B' is a lower bound on the optimal solution to B.

The solution process consists of solving B' and obtaining $(\tilde{\underline{x}}_0^0, \tilde{y}_i^0)$ and $\hat{z}^0 = \underline{c}_0 \tilde{\underline{x}}_0^0 + \sum_{i=1}^{m} \tilde{y}_i^0$, (if B' is infeasible then it follows that B is infeasible). The trial solution $\tilde{\underline{x}}_0^0$ is then replaced in B and the problem is solved for the values of y_i. Solving B for a given $\underline{x}_0 = \tilde{\underline{x}}_0^0$ is equivalent to solving the subprograms D_i for that value of $\underline{x}_0$ and obtaining a set of extreme points $\hat{\underline{u}}_{ik}$ and a set of values y_i^0. Thus problem B for $\underline{x}_0 = \tilde{\underline{x}}_0^0$ has the solution $(\tilde{\underline{x}}_0^0, y_i^0)$, and $z = \underline{c}_0 \tilde{\underline{x}}_0^0 + \sum_{i=1}^{m} y_i^0$. Now z and $\hat{z}^0$ are compared. If $\hat{z}^0 < z$, then the current constraints of B' do not determine optimality; new constraints generated from the extreme points $\hat{\underline{u}}_{ik}$ obtained from the solution to D_i are therefore added to B' to complete one iteration of the algorithm. On the other hand, if $\hat{z}^0 \geq z$ the solution $\hat{z}^0$ to B' is optimal for B, and thus it satisfies the original problem A.

Next we shall restate the algorithm, considering all of the different situations. For a rigorous proof of the termination rules, the reader is referred to the work of Benders, [1].

INITIAL STEP. Obtain a subset of extreme points K_i' and/or extreme rays L_i' to generate problem B'.

STEP a. Solve problem B'

a.1) If B' is infeasible, B is infeasible; stop, problem A has no feasible solution

a.2) If B' is unbounded below, take as the value of $\underline{x}_0$ for step b any feasible $\tilde{\underline{x}}$ of B' corresponding to a small value of $\hat{z}$.

a.3) Otherwise B' has an optimal solution $\hat{z}^0$ and $(\tilde{\underline{x}}_0^0, \tilde{y}_i^0)$, so go to step b

STEP b. Solve problem B for $\underline{x}_0 = \tilde{\underline{x}}_0^0$ from step a. That is, solve all subprograms D_i for that value of $\underline{x}_0$.

b.1) If one of the subprograms D_i is infeasible, stop. Problem A is either infeasible or unbounded below. (This case may occur only during the first iteration.)

b.2) For each subprogram D_i that is unbounded above along the half line $\{\underline{u}_i \,/\, \underline{u}_i = \hat{\underline{u}}_{ik} + \delta\, \bar{\underline{u}}_{i\ell}\}$, add for $\bar{\underline{u}}_{i\ell}$ a constraint of the form (A.25) to problem B'. If also $(\underline{b}_i - A_i\, \tilde{\underline{x}}_o^0)'\, \hat{\underline{u}}_{ik} > \tilde{y}_i^0$, then $\hat{\underline{u}}_{ik}$ defines a constraint of the form (A.24) to be added to B'. Go to step a.

b.3) Otherwise all D_i have an optimal solution $\hat{\underline{u}}_{ik}^0$. Calculate $y_i^0 = (\underline{b}_i - A_i\, \tilde{\underline{x}}_o^0)'\, \hat{\underline{u}}_{ik}^0$ and go to step c.

STEP c. Obtain $z = \underline{c}_o\, \tilde{\underline{x}}_o^0 + \sum_{i=1}^{m} y_i^0$

c.1) If $\tilde{z}^0 = z$, stop; the solution z and $(\tilde{\underline{x}}_o^0, y_i^0)$ is optimal for B. By obtaining the solutions to the problems P_i we obtain $\underline{x}_i^0$, and thus z and $(\tilde{\underline{x}}_o^0, \underline{x}_i^0)$ constitute an optimal solution for A.

c.2) If $\tilde{z}^0 < z$ then each of the $\hat{\underline{u}}_{ik}^0$ defines a new constraint of the form (A.24) to be added to B'. Go to step a.

We note the following properties of the algorithm:

i) Each time that step a is executed (i.e., problem B' solved), $\tilde{z}^0$ constitutes a lower bound on the optimal solution z^0 which is also a better lower bound than that of the previous iteration.

ii) Whenever step b is executed (i.e., problem B is solved for $\underline{x}^0 = \tilde{\underline{x}}_o^0$), either we obtain the *optimal* solution to B, (detected by c.1) or the solution to B constitutes an upper bound on z^0, (i.e., $z^0 \le \underline{c}_o\, \tilde{x}_o^0 + \sum_{i=1}^{m} y_i^0$). The best upper bound, however, does not necessarily correspond to the value of the objective function of B obtained in the current iteration, but is obtained as the minimum value of the objective function of B over all iterations performed so far.

A.6 THE CASE WHERE $\tilde{S}_0$ IS THE SET OF VERTICES OF THE UNIT HYPERCUBE

In this section we consider problem B in a slightly different form, in order to conveniently study the case where S_0 is given by expression (A.7).

In expression (A.11), set $y_0 = \underline{c}_0\, \underline{x}_0 + \max_{k\epsilon K_i} \{\sum_{i=1}^{m} (\underline{b}_i - A_i\, \underline{x}_0)'\, \hat{u}_{ik}\}$;

then for any combination of m extreme points taken one from each convex set S_i, the following condition holds:

$$\underline{c}_0\, \underline{x}_0 + \sum_{i=1}^{m} (\underline{b}_i - A_i\, \underline{x}_0)'\, \hat{\underline{u}}_{ik} \leq y_0 \;;\; k \in K_i$$

Expression (A.11) may therefore be conveniently expressed in the form of the equivalent problem to A:

$\bar{B}$: Minimize $\quad z = y_0$ (A.26)

Subject to $\quad \underline{c}_0\, \underline{x}_0 + \sum_{i=1}^{m} (\underline{b}_i - A_i\, \underline{x}_0)'\, \hat{\underline{u}}_{ik} \leq y_0$,

$\qquad , i=1,\ldots,m;\; k \in K_i$ (A.27)

$(\underline{b}_i - A_i\, \underline{x}_0)'\, \bar{\underline{u}}_{i\ell} \leq 0\;,\; i=1,\ldots,m;\; \ell \in L_i$ (A.28)

$A_0\, \underline{x}_0 = \underline{b}_0$ (A.29)

$x_{0j} = 0$ or 1 (A.30)

y_0 unrestricted (A.31)

The (0-1) mixed-integer linear programming problem A is thus equivalent to $\bar{B}$. Except for the unrestricted variable y_0, problem $\bar{B}$ is a (0-1) all-integer programming problem with n_0 integer variables and a potentially large number of constraints. For certain network flow problems involving decision variables and satisfying some additional integrality conditions on the input data, the variable y_0 may also be restricted to be integer (cf. Chapter V). It is for this class of problems that we shall discuss the solution procedure of step a of the Benders algorithm.

The problem $\bar{B}'$, ($\bar{B}$ with a subset of (A.27) and (A.28), will then contain $n_0 + 1$ integer variables and may be solved by means of a branch and bound algorithm (cf. Chapter II). However, we consider that the primal all-integer algorithm, developed independently by Young [4] and González [5] and denoted here as the *Young-González algorithm*, is more suited to the conditions of the problem. Indeed, the following properties indicated by González [5] fully apply to the solution of $\bar{B}'$:

i) González presents a special procedure for treating unsigned variables, which applies in our case to the integer and unrestricted variable y_0 in $\bar{B}'$.

ii) He also presents a special way to handle integer variables restricted to take on the values 0 or 1.

iii) Given the nature of the objective function, (i.e., y_0 is the only variable and has a positive coefficient), the initial tableau already satisfies optimality (first row elements ≤ 0) although it may not be primal feasible. Therefore it suffices to apply the González procedure to obtain a starting feasible solution.

Finally, we have observed that each time a new constraint is added to a problem after step b. of the Benders algorithm, problem $\bar{B}'$ may be reoptimized simply by updating the constraint in terms of the current tableau. Since the slack of the constraint will be negative, and it is restricted to be positive, we apply the González method to remove the infeasibility of the slack variable. This operation may alter the optimality of the first row. If this is the case, the tableau is then reoptimized. From the properties we have indicated, we consider that the application of the algorithm in Chapter V for the solution of the multi-stage link addition problem is justified.

A.7 SOLUTION OF AN EXAMPLE PROBLEM

Consider the following problem:

$$A : \text{Min } z = 7x_1 + 6x_2 + 5x_3 + 4x_4 + 3x_5 - 12x_6$$

$$4x_1 + 2x_2 + 5x_3 \geq 1$$

$$4x_1 + 3x_2 + x_3 \geq 8, \quad \forall\, x_i \geq 0$$

$$5x_1 + x_4 - 3x_6 \geq 5$$

$$2x_1 + x_5 - 4x_6 \geq 6$$

Although the problem is not in standard form, we shall not add slack variables, but will use the graphical representation of the sets S_i. Consider two subprograms S_1 and S_2, indicated in Figs. A-1 and A-2 respectively. Then the various elements of problem A are

$$A_1 = \begin{bmatrix} 4 \\ 4 \end{bmatrix},\ A_2 = \begin{bmatrix} 5 \\ 2 \end{bmatrix},\ B_1 = \begin{array}{cc} x_2 & x_3 \\ \multicolumn{2}{c}{\begin{bmatrix} 2 & 5 \\ 3 & 1 \end{bmatrix}} \end{array},\ B_2 = \begin{array}{ccc} x_4 & x_5 & x_6 \\ \multicolumn{3}{c}{\begin{bmatrix} 1 & 0 & -3 \\ 0 & 1 & -4 \end{bmatrix}} \end{array},\ \underline{b}_1 = \begin{bmatrix} 1 \\ 8 \end{bmatrix},\ \underline{b}_2 = \begin{bmatrix} 5 \\ 6 \end{bmatrix}$$

$$S_0 = \bar{S}_0 = \{x_1 \,/\, x_1 \geq 0\}$$

$$S_1 = \{(u_1, u_2) \,/\, 2u_1 + 3u_2 \leq 6;\ 5u_1 + u_2 \leq 5;\ u_1, u_2 \geq 0\}$$

$$S_2 = \{(u_3, u_4) \,/\, u_3 \leq 4;\ u_4 \leq 3;\ -3u_3 - 4u_4 \leq -12;\ u_3, u_4 \geq 0\}$$

INITIAL STEP We shall assume that an extreme point for each S_i is known, say $\hat{\underline{u}}_{11} = (0, 0)$ and $\hat{\underline{u}}_{21} = (0, 3)$.

ITERATION 1

STEP a. We solve B'. To reduce the number of variables in B' to 2, (i.e., $\tilde{x}_1$ and $\tilde{y}$), thus permitting a graphical solution of this step, we shall set $y = \underline{c}_0\, \underline{x}_0 + \sum_{i=1}^{2} \max_{k \in K_i} (\underline{b}_i - A_i\, \underline{x}_0)'\, \hat{\underline{u}}_{ik}$.

$$\begin{aligned} B' : \ &\text{Min } \tilde{z} = \tilde{y} \\ &\tilde{y} - \tilde{x}_1 \geq 18 \\ &\tilde{x}_1 \geq 0 \quad \tilde{y} \text{ unrestricted} \end{aligned}$$

By inspection (see Fig. A-3) the optimal solution is $\tilde{z}^0 = 18$, and $\tilde{x}_1^0 = 0$.

STEP b. We solve the subprograms D_1 and D_2 for $x_1 = \tilde{x}_1^0 = 0$.

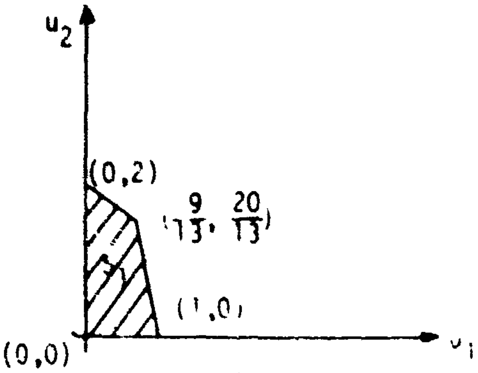


FIG. A-1

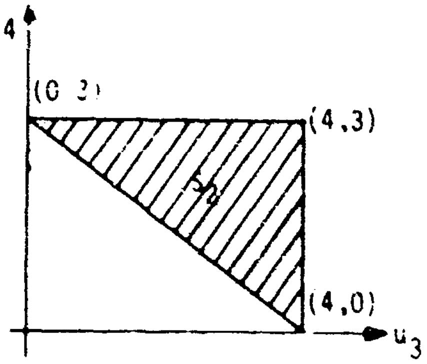


FIG. A-2

$$D_1 : \text{Max } z_1 = u_1 + 8u_2 \qquad D_2 : \text{Max } z_2 = 5u_3 + 6u_4$$

$$\begin{aligned} 2u_1 + 3u_2 &\le 6 \\ 5u_1 + u_2 &\le 5 \\ u_1, u_2 &\ge 0 \end{aligned} \qquad \begin{aligned} u_3 &\le 4 \\ u_4 &\le 3 \\ -3u_3 - 4u_4 &\le -12 \\ u_3, u_4 &\ge 0 \end{aligned}$$

D_1 : optimal solution $z_1^0 = 16$, $\underline{u}_{12} = (0, 2)$

D_2 : optimal solution $z_2^0 = 38$, $\underline{u}_{22} = (4, 3)$

STEP c. We obtain z for the current value of x_1: $z = 7.\ \bar{x}_1^0 + z_1^0 + z_2^0 = 54$;

therefore $\bar{z} = 18 < z = 54$, and the optimal solution z^0 is bounded as follows: $18 \le z^0 \le 54$

ITERATION 2

STEP a. B' : Min $\bar{z} = \bar{y}$

$$\bar{y} - \bar{x}_1 \ge 18$$

$$\bar{y} + 23\,\bar{x}_1 \ge 54 \quad \text{(new constraint)}$$

$$\bar{x}_1 \ge 0,\ \bar{y} \text{ unrestricted}$$

Optimal solution, (see Fig. A-4): $\bar{z}^0 = 19.5$, and $\bar{x}_1^0 = 1.5$

STEP b. We solve D_1 and D_2 for $x_1 = \bar{x}_1^0 = 1.5$

$$D_1 : \text{Max } z_1 = -5u_1 + 2u_2 \qquad D_2 : \text{Max } z_2 = -2.5u_3 + 3u_4$$

$$(u_1, u_2) \in S_1 \qquad (u_1, u_2) \in S_2$$

D_1 : optimal solution $z_1^0 = 4$, $\underline{\hat{u}}_{12} = (0, 2)$

D_2 : optimal solution $z_2^0 = 9$, $\underline{\hat{u}}_{21} = (0, 3)$

STEP c. $z = 7\bar{x}_1^0 + z_1^0 + z_2^0 = 23.5$; therefore $\tilde{z}^0 = 19.5 < z = 23.5$ and the updated bounds for z^0 are $19.5 \leq z^0 \leq 23.5$.

ITERATION 3

STEP a. B' : Min $\tilde{z} = \tilde{y}$

$$\tilde{y} - \tilde{x}_1 \geq 18$$

$$\tilde{y} + 23\tilde{x}_1 \geq 54$$

$$\tilde{y} + 7\tilde{x}_1 \geq 34 \text{ (new constraint)}$$

$$\tilde{x}_1 \geq 0,\ \tilde{y} \text{ unrestricted}$$

Optimal solution, (see Fig. A-5): $\tilde{z}^0 = 20$ and $\tilde{x}_1^0 = 2$

STEP b. We solve D_1 and D_2 for $x_1 = \tilde{x}_1^0 = 2$

D_1 : Max $z_1 = -7u_1$ $\quad$ $(u_1, u_2) \in S_1$

D_2 : Max $z_2 = -5u_3 + 2u_4$ $\quad$ $(u_1, u_2) \in S_2$

D_1 : Optimal solution $z_1^0 = 0$, and either $\underline{\hat{u}}_{11} = (0,0)$ or $\underline{\hat{u}}_{12} = (0,2)$

D_2 : Optimal solution $z_2^0 = 6$, $\underline{\hat{u}}_{21} = (0,3)$

STEP c: $z = 7\tilde{x}_1^0 + z_1^0 + z_2^0 = 20$, and $\tilde{z}^0 = z$, optimality.

Thus the optimal solution is: $z^0 = 20$, $x_1^0 = 2$, $u_1^0 = 0$, $u_2^0 = 0$ or 2 $u_3^0 = 0$, $u_4^0 = 3$.

To obtain the optimal values of the primal variables x_2, x_3, x_4, x_5 and x_6, it suffices to solve the linear programs P_1 and P_2 for $x_1 = x_1^0 = 2$

P_1 : Min $z_1 = 6x_2 + 5x_3$

$$2x_2 + 5x_3 \geq -7$$

$$3x_2 + x_3 \geq 0$$

P_2 : Min $z_2 = 4x_4 + 3x_5 - 12x_6$

$$x_4 - 3x_6 \geq -5$$

$$x_5 - 4x_6 \geq 2$$

P_1 : Optimal solution, $z_1^0 = 0$, $x_2^0 = x_3^0 = 0$

P_2 : Optimal solution, $z^0 = 6$, $x_4^0 = 0$, $x_5^0 = \frac{26}{3}$, $x_6^0 = 5/3$

Normally this step of solving P_1 and P_2 is avoided since, for most algorithms, the solution to problems D_1 and D_2 also determine the optimal solution to their dual programs P_1 and P_2.

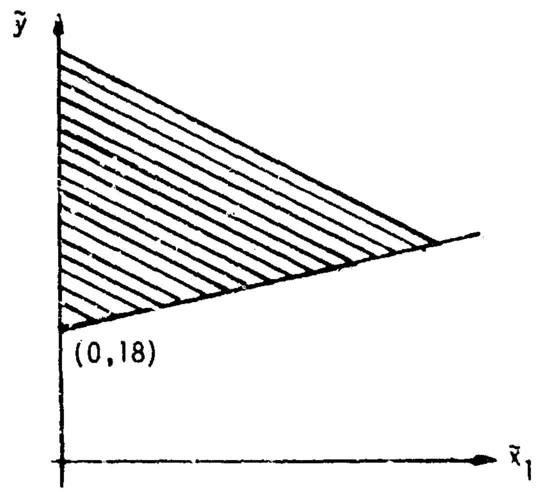


FIG. A-3

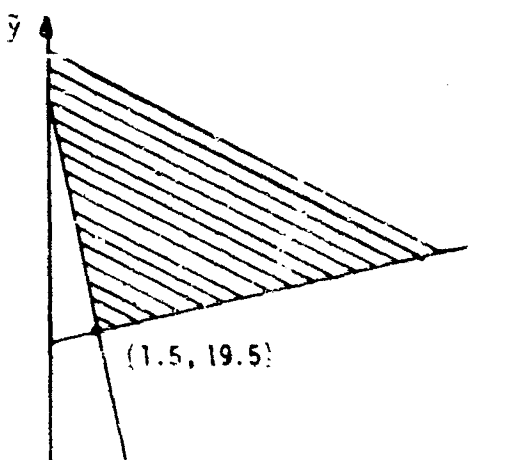


FIG A-4

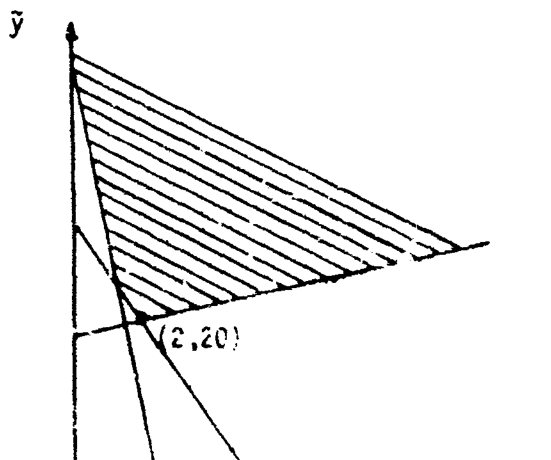


FIG. A-5

A.8 NOTES TO APPENDIX A

[1] Benders, J. F., "Partitioning Procedures for Solving Mixed-Variables Programming Problems", *Numerische Mathematik*, Vol. 4, 1962, pp. 238-252.

[2] Balinski, M. L., "Integer Programming: Methods, Uses, Computations", *Management Science*, Vol. 12, No. 3, November 1965, pp. 253-313.

[3] Dantzig, G. B. and P. Wolfe, "A Decomposition Principle for Linear Programs", *Operations Research*, Vol. 8, No. 1, 1960, pp. 101-111.

[4] Young, R., "A Primal (all-integer) Integer Programming Algorithm", *Journal of Research of the National Bureau of Standards* - B. Mathematics and Mathematical Physics, Vol. 69B, No. 3, July-September, 1965, pp. 213-250.

[5] González-Zubieta, R. H., *On Some Aspects of Integer Linear Programming*, Technical Report No. 16, Operations Research Center, Massachusetts Institute of Technology, June 1965.

APPENDIX B

HISTORICAL SURVEY OF OPTIMIZATION THEORY

B.1 EARLY DEVELOPMENTS

It is now generally accepted that the first investigations of optimization were carried out by the Greek geometricians of the first Alexandrian School (circ. 300-30 B.C.). The well-known historian of mathematical thought, Moritz Cantor [1], attributes to Euclid (Book VI, prop. 27) the first example of an extremal problem in the history of Mathematics. The proposition proves by synthetic means that if a straight line segment is divided into two parts the product of both parts is maximum when the parts are equal. No less familiar to Euclid were the following problems: "The perpendicular is the minimum among all straight lines that may be drawn from a point to a line." and "The diameter of a circle is the maximum among all inscribed lines."

The other two geometers who share with Euclid the fame accredited to the Greeks of that school are Archimedes and Apollonius of Perga, who also were concerned with problems of maxima and minima. The former, in the second book of his work on the *Sphere and Cylinder*, proposes the following: "of all spherical segments whose surfaces are equal the hemisphere has the greatest volume". The latter, celebrated for his work on the conic sections, determines in his fifth book "the shortest line that may be drawn from a point to a given conic section".

Pappus of Alexandria, who belongs to the second Alexandrian school (30 B.C. - 641 A.D.) is credited with the solution of several "isopherimetric problems". The first ten propositions of his fifth book, [3], are directed towards the proof of the proposition that among all figures of same perimeter, the circle has the greatest area. He later remarks that if most of the properties of the sphere had already been found, one remained to be proven: "of all the solid figures having the same area the sphere has the greatest volume". His proof is not general and the problem became one of the most controversial issues of mathematical history. Its rigorous solution was not obtained until the second half of the eighteenth century by means of the calculus of variations.

In his book VII, Proposition 61, Pappus solves, by synthetic geometry, the following problem: "Minimize the function $\frac{x(a-x)}{(x-a_1)(a_2-x)}$"; his solution is much simpler and more elegant than its analytic counterpart using the differential calculus.

In the seventeenth century, after a deep lapse of mathematical progress characteristic of the Middle Ages, and before Newton and Leibnitz developed the calculus, Fermat published his *Methodus ad disquirendum maximum et minimum*. Ball [2] suggests that his method was developed after a remark by Kepler, that the values of a function in the neighborhood of an extreme point on either side must be equal. He solves the problem treated by Euclid of finding two numbers such that its sum is given and its product is to be minimized. His method is equivalent to taking a neighboring value of x, namely x + e, where e is very small, and setting $x(a-x) = (x+e)(a-x-e)$. Simplifying the algebra and ultimately setting $e = 0$, the solution is obtained for $x = \frac{a}{2}$. Later Huygens, from the Hague, stated in general terms the rule used by Fermat. About 1673 he solved the problem: two points P_1, P_2 not on the straight line AB are given. Find a point P on AB such that $\overline{PP}_1^2 + \overline{PP}_2^2$ is a minimum.

B.2 CLASSICAL PERIOD

The epoch of formal development of classical optimization theories (indirect methods based on the differential calculus) begins with the invention of the calculus. The theories obtain necessary conditions to be satisfied by an optimum point. Sufficiency was seldom satisfied and new means to prove it remained to be discovered. The main contributors were Newton, who applied his *method of fluxions* to problems of maxima and minima, and Leibnitz, who published in his *Acta Eruditorum* of October 1684 a general method for finding maxima and minima.

During the second part of the eighteenth century a large class of optimization problems, the optimization of a definite integral, was studied by the Bernoullis and Euler and systematized by Lagrange. This new branch of optimization theory was termed the *Calculus of Variations*, a name suggested by Euler. Previous developments in mechanics suggested to Euler that all natural phenomena present extrema, and his later work constitutes an important application of optimization theory to mechanical systems. A complete account of optimization problems in mechanics can be found in [10]. On the solution of optimization problems subject to subsidiary conditions, a systematic method was given by Lagrange in his *Théorie des Fonctions*, which determines a set of necessary conditions for an extremum of a function subject to equality constraints.

In the nineteenth century the work of Weierstrass of the University of Berlin served to formalize the theory of maxima and minima. He was primarily concerned with existence conditions, which had been somewhat disregarded probably due to the fact that in many physical applications either a maximum or a minimum obviously exists. His existence theorem, based on the work of Bolzano, states that: *if a function $f(x)$ is continuous in $a \le x \le b$, there exists ξ_1 and ξ_2, $a \le \xi_1 \le b$, $a \le \xi_2 \le b$, for which the function attains its largest value M and its smallest value m.*

Jacob Steiner, another mathematician of the University of Berlin, representative of the geometric school, solved in the early nineteenth century a problem posed earlier by Fermat, which has had important applications in generalized form to location theory. The problem is: given three points A B C in a plane, find a fourth point P such that the sum of the Euclidian distances from each of the three points to P is minimized. This problem has been widely publicized by Courant and Robbins [11] and has lately been treated by Kuhn in [12] to present an interesting duality concept in nonlinear programming.

Another source of solution methods for optimization problems which has proved effective is the general theory of inequalities (see for example [6] or [15]). It is worth mentioning that the application is reciprocal: namely, inequality theorems may be proved with the auxiliary solution of a maximum and minimum problem, while certain optimization problems may be solved by the use of known inequalities. This reciprocal character is formalized by Chrystal in [5].

An inequality that has largely contributed to the method mentioned above and that has been a basic cornerstone of the latest development of optimization theory, Geometric Programming, [7], is the so-called *geometric inequality*. This inequality states that for a finite set of non-negative numbers, the arithmetic mean is at least as great as the geometric mean. The English mathematician MacLaurin is credited with the first general proof of the geometric inequality. He enunciated the theorem in the following geometric form, [8]: *If a line AB is divided into any number of parts, the product of all those parts will be a maximum when the parts are equal among themselves.* The best known analytical proof of this classical inequality, however, is due to Cauchy, [16].

In the work of Harris Hancock [12], [13] published in 1917, we find an excellent summary of what may be considered the state of classical optimization theory up to that time. In [13], Hancock indicates that several inaccuracies carried through from the developments of Lagrange were corrected when a major revision of the theory of maxima and minima, suggested by Peano of Turin, was carried through by the work of Scheeffer, Stolz and Dontscher. In [13], sections 109-112, he presents the treatment of constrained optimization problems subject to inequality restrictions, and makes use of quadratic slack variables to reduce the problem to equality constraints.

B.3 MODERN PERIOD: FIRST DECADE

It is during the modern period that the theory of maxima and minima has been widely broadened and given the now generally accepted name of *Optimization Theory*. Primarily responsible for such a task are the American scientists who carried on the development of the theory during and after World War II.

The modern period of optimization theory (or "renaissance", as Nemhauser [17] likes to put it), started in 1947 with G. Dantzig's Simplex method for the solution of linear programs. In the two decades since that event, the development of optimization theory has been extremely fruitful in both pure analytical techniques and applications to the managerial sciences, the military, engineering, and the physical sciences.

As in various other fields, the substantial new developments that took place in the middle fifties are due in great part to the advent of the digital computer as a common tool in scientific research and development. This fact may very well lead in the future to the study of the history of sciences by dividing it into two parts: before the computer and after the computer. This is no less true in the case of optimization theory.

The first decade of the modern period, 1947-1957, is characterized by the formal solution of the linear programming problem and the rigorous analysis of its underlying mathematical theory. The work performed during the two years 1947-1949 was presented at the now historic *Cowles Commission conference* in Chicago in 1949, and selected papers were published in *Activity Analysis of Production and Allocation*, edited by T. C. Koopmans.

A number of applications in business and industry followed, associated with the names of Charnes and Cooper, who published with Henderson in 1953 what constitutes the first textbook on the subject matter [18]. The book of Gass [19], although published in 1958, may also be considered a product of the early developments of linear programming.

The principles of the mathematical theory, as well as the statement of duality, were laid down by von Neumann. The rigorous studies on duality and linear inequality theories were carried out and published in the work edited by H. Kuhn and A. Tucker of the Princeton school in 1956, *Linear Inequalities and Related Systems*.

The success and achievements of this decade stem largely from the development of computer codes for the solution of linear programs which bridge the gap between theory and practice and open a wide avenue of applications.

Almost in parallel with linear programming, R. Bellman [20], S. Dreyfus [15] and others have developed another powerful optimization technique, *dynamic programming*, of particular application to problems of optimal control and multistage decision processes.

For a complete account of the background of and contributors to the modern development of mathematical programming, the reader is referred to Dantzig's own account, [21].

B.4 MODERN PERIOD SECOND DECADE

The second decade of the modern period has been more prolific by far in the development of new methodologies of optimization It has seen the verification of Dantzig's prediction in his opening address to the Third Symposium on Mathematical Programming held in Santa Monica, California, in 1959, [22]: "Today, we who are gathered here are about to witness the start of an explosion "

We shall mention briefly the highlights of such accomplishments and the principal contributors to each field.

The special *unimodularity property* of certain classes of linear programs observed by Dantzig [22], has been a keystone of the development of network flow theory The principal contributors have been Ford and Fulkerson [23], [24] who proved the maximum-flow-minimum cut theorem for homogeneous commodity flow in networks; Berge [25] with his rigorous work on graph theory; and Kuhn [26] with his work on combinatorics and the assignment problem Generalizations of network flow theory have been made by Gomory and Hu [27] on multi-terminal flows and by Jewell [28] on multi-commodity flow problems

In discrete and integer programming, this decade has seen the systematic development of cutting plane methods by Gomory [29], and the so-called *branch and bound* techniques by Land and Doig [31], Little et al. [30], and Balas [71], (cf Chapter II) For detailed information on the subject, the reader is referred to the excellent work of Balinski [32] which constitutes an exhaustive survey of integer programming

Pressed perhaps by the growing number of applications with the ever-increasing sizes of problems, particular attention was given, starting around 1959, to the exploitation of special structures presented by certain classes of problems From these studies evolved the Decomposition Principle of Dantzig and Wolfe [33], without a doubt, a major contribution to the operational solution of linear programs Other types of partitioning algorithms have been proposed by Benders [34], Balas [69], Rosen [70], and others

In the area of stochastic programming, initiated by the two-stage model of Dantzig [35] and the work of Tintner [36], much remains to be investigated The last ten years have witnessed the work of Charnes and

Cooper and their chance constrained model [37] as well as the most important work of Madansky [38], [39]. Of special interest in the last few years is the work of Dantzig [40], Van Slyke [41], and Wets [42] on the integration of mathematical programming and optimal control theory, and the application of the two-stage approach of stochastic programming to the latter. As Madansky [43] puts it, the *risk* introduced into the programming problem has to do with the probability distribution of the random variables of the problem when these are completely known. The *uncertainty* arises when the probability distribution is known in form but one or more parameters are unknown An important aspect likely to be developed in the future is the introduction of Bayesian concepts in the multistage models, providing the capability of updating the probability distributions associated with the problem as more information is available in the process.

The remaining topic, and its basic theoretical paper by Kuhn and Tucker in 1951, [44], generalizing Lagrange's method for the case of inequality constraints, is the topic of nonlinear programming. Approximate solution methods were developed during the first decade by Charnes and Lemke [45] and Dantzig [46] The special case of quadratic programming has been well-studied by Beale in 1955 [47], Frank and Wolfe [48], and Wolfe [49]. Other solution techniques of classical nature known as gradient methods dating back to Cauchy, were consolidated in the early book edited by Arrow, Hurwicz and Uzawa [50] and the later work of Lemke [51], Rosen [52], Zoutendijk [53], Davidon [54], Doerfler [55], and others.

From the field of numerical analysis several methods for unconstrained optimization have been developed in the sixties, and in several instances they have been generalized for handling constraints Of the indirect optimization type we mention the work of Fletcher and Reeves [56]. The direct search methods are based generally on the work of Hooke and Jeeves [58], and the random search methods on the work of Karnopp [59] and Brooks [60].

As a final remark on nonlinear programming, we mention again the latest development that seems to be a very promising optimization tool for engineering design, constituting a generalization of the use of inequalities in the solution of extremum problems The work has been given the name of *Geometric Programming* by its developers, Zener, Duffin and

Peterson, 1967, [7], and it deals with the optimization of unconstrained or constrained "posynomials" (positive polynomials). It has already been generalized for the case of negative terms by Passy and Wilde, [61].

In all, the difficult field of nonlinear programming has not yet yielded to a systematic treatment; we feel that a unifying theory remains to be presented.

Of the text books of the second decade, we mention the two books of Hadley in linear and nonlinear programming [62], [63]. The latter, if perhaps not a complete or perfectly organized work, is the first general text in this area. The book of Dantzig [21], that of Vadja [64], and probably the best text so far in linear programming, the translation by Jewell of Simmonard's textbook [65], also were published in this period. Finally, we mention the book of Wilde and Beightler [61] and the book on nonlinear programming edited by Abadie [67].

B.5 FUTURE RESEARCH

The Sixth International Symposium on Mathematical Programming that took place at Princeton in August 1967, marks the beginning of the third decade of research and development on optimization theory. From the work presented there, it is possible to infer which are the currents of research likely to be developed in the near future. Although substantial research seems to be underway in most areas of optimization theory, we feel that special effort is being devoted to the following areas of research.

The field of discrete linear programming is very likely to develop rapidly, as it is now provided with a duality theory analogous to its continuous counterpart, developed by Balas [72]. Also, important contributions have been made by Balinski [73] on a pair of related problems known as *the maximum match and the minimum covering problems*. Primal-dual methods are therefore likely to be developed which might be of special use, for example, in network flow theory for problems involving networks with disjunctive arcs (i.e., flow either zero or at upper bound). Such network models are particularly suited for solving the class of problems treated in Chapters III and IV. The author is currently engaged in this specific problem.

In nonlinear programming, more and more applications in the context of engineering design seem likely. Also, theoretical extensions of geometric programming such as the one presented by Avriel and Wilde [74] on stochastic geometric programming, may be expected. The same may be said about the important topic of control theory. Finally, we feel that the efficient exploitation of highly-structured optimization models will necessarily lead to new schemes for solution of large-scale problems.

To close this appendix we shall mention that the development of integrated optimization systems, employing new computer technology and the wealth of optimization techniques currently available, holds great promise in the solution of large-scale optimum design problems. The need for powerful synthesis algorithms such as those mentioned in the introductory chapter of this work will contribute to this development.

B.6 NOTES TO APPENDIX B

[1] Cantor, M., *Vorlesungen über Geschichte der Mathematik*, Vol. I, Leipzig, 1900, p. 252.

[2] Rouse Ball, W. W., *A Short Account of the History of Mathematics*, Dover Publications, Inc., New York, 1960.

[3] Ver Eecke, P., *Pappus D'Alexandrie, la Collection Mathématique*, Desclée de Bouwer et Cie., Paris, 1933.

[4] Heath, T. L., *The Thirteen Books of Euclid's Elements*, Vol. II, Dover Publications, Inc., New York, 1956.

[5] Chrystal, G., *Algebra*, Part II, A. and C., Black, London, 1900, pp. 52-55.

[6] Hardy, G. H., J. E. Littlewood, and G. Polya, *Inequalities*, Cambridge University Press, England, 1959.

[7] Duffin, R. J., E. L. Peterson, and C. M. Zener, *Geometric Programming*, John Wiley & Sons, New York, 1967.

[8] MacLaurin, C., "Concerning the roots of equations, with the demonstration of other rules in algebra", *Phil. Transactions*, 36, pp. 59-96, 1729.

[9] Bell, E. T., *The Development of Mathematics*, McGraw-Hill, Inc., New York, 1940.

[10] Castigliano, C. A. P., *The Theory of Equilibrium of Elastic Systems and its Applications*, Dover Publications, Inc., 1967.

[11] Courant, R., and H. Robbins, *What is Mathematics?*, Oxford University Press, New York, 1941.

[12] Kuhn, H. W., "On a Pair of Dual Nonlinear Programs", in *Nonlinear Programming*, J. Abadie (ed.), Interscience Publishers, John Wiley, Inc., New York, 1967.

[13] Hancock, H., *Theory of Maxima and Minima*, Dover Publications, Inc., 1960.

[14] ________, *Lectures on the Calculus of Variations*, Bulletin of Mathematics, No. 1, University of Cincinnati, Ohio, 1904.

[15] Beckenbach, E. and R. Bellman, *An Introduction to Inequalities*, Random House, 1961

[16] Cauchy, A. L., *Cours d'Analyse de L'Ecole Royale Polytechnique*, (Analyse Algébrique), De L'Imprimerie Royale, 1821, pp. 375-377.

[17] Nemhauser, G., *Introduction to Dynamic Programming*, John Wiley & Sons, Inc., New York, 1966.

[18] Charnes, A., W. W. Cooper, and A. Henderson, *An Introduction to Linear Programming*, John Wiley & Sons, Inc., New York, 1953.

[19] Gass, S. I., Linear Programming: *Methods and Applications*, McGraw-Hill Book Co., Inc., New York, 1958.

[20] Bellman, R., *Dynamic Programming*, Princeton University Press, Princeton, New Jersey, 1957.

[21] Dantzig, G. B., *Linear Programming and Extensions*, Princeton University Press, Princeton, New Jersey, 1963.

[22] Dantzig, G. B., "Application of the Simplex Method to a Transportation Problem", in T. C. Koopmans (ed.), *Activity Analysis of Production and Allocation*, John Wiley & Sons, Inc., New York, 1951, pp. 359-373.

[23] Ford, L. R., and D. R. Fulkerson, "Maximal Flow Through a Network", The RAND Corporation, Research Memorandum RM-1400, November 19, 1954.

[24] __________, *Flows in Networks*, The RAND Corporation, Report R-375, December 20, 1960. Published by Princeton University Press, New Jersey, 1962.

[25] Berge, C., *Théorie des Graphes et ses Applications*, Dunod, Paris, 1958.

[26] Kuhn, H. W., "The Hungarian Method for the Assignment Problem", *Naval Res. Logist. Quart.*, Vol. 2, 1955, pp. 83-97.

[27] Gomory, R. E., and T. C. Hu, "Multi-terminal Network Flows", I.B.M. Research Report, 1960.

[28] Jewell, W., "Optimal Flow Through Networks", Interim Technical Report No. 8, Massachusetts Institute of Technology, Cambridge, (Mass.), 1958.

[29] Gomory, R. E., "Essentials of an Algorithm for Integer Solutions to Linear Programs", *Bull. Amer. Math. Soc.*, Vol. 64, No. 5, 1958.

[30] Little, J. D. C., K. G. Murty, D. W. Sweeney, and C. Karel, "An Algorithm for the Traveling Salesman Problem", *Operations Research*, Vol. 11, 1963, pp. 972-989.

[31] Land, A. H. and A. G. Doig, "An Automatic Method of Solving Discrete Programming Problems", *Econometrica*, Vol. 28, 1960, pp. 497-520.

[32] Balinski, M. L., "Integer Programming: Methods, Uses, Computation", *Management Science*, Vol. 12, No. 3, November 1965, pp. 253-313.

[33] Dantzig, G. B. and P. Wolfe, "A Decomposition Principle for Linear Programs", RAND Paper P-1544, November 1958.

[34] Benders, J. F., "Partitioning Procedures for Solving Mixed-Variables Programming Problems", *Numerische Mathematik*, Vol. 4, 1962, pp. 238-252.

[35] Dantzig, G. B., "Linear Programming under Uncertainty", *Management Science*, Vol. 1, 1955, pp. 197-206.

[36] Tintner, G., "Stochastic Linear Programming with Application to Agricultural Economics", *Second Symposium on Linear Programming*, Nat. Bureau of Standards, Washington, D. C., Vol. 1, 1955, pp. 197-228.

[37] Charnes, A. and W. W. Cooper, "Chance-constrained Programming", *Management Science*, Vol. 6, 1959, pp. 73-79.

[38] Madansky, A., "Some Results and Problems in Stochastic Linear Programming", The RAND Corporation, paper P-1596, January 19, 1959.

[39] ________, "Inequalities for Stochastic Linear Programming Problems", *Management Science*, Vol. 6, 1960, pp. 197-204.

[40] Dantzig, G. B., "Linear Control Processes and Mathematical Programming", J. SIAM, Vol. 4, No. 1, 1966, pp. 56-60.

[41] Van Slyke, R., "Programming Under Uncertainty and Stochastic Optimal Control", J. SIAM Control, Vol. 4, No. 1, 1966, pp. 179-193.

[42] Wets, J. B., "Programming under Uncertainty: The Equivalent Convex Program", J. SIAM Appl. Math., Vol. 14, No. 1, 1966, pp. 89-105.

[43] Madansky, A., "Linear Programming under Uncertainty", in *Recent Advances in Mathematical Programming*, R. L. Graves and P. Wolfe (eds.), McGraw-Hill Book Company, Inc., New York, 1963.

[44] Kuhn, H. W. and A. W. Tucker, "Nonlinear Programming", in J. Neyman (ed.), *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, U. of California Press, Berkeley, Calif., 1951, pp. 481-492.

[45] Charnes, A. and C. Lemke, "Minimization of Nonlinear Separable Convex Functionals", *Naval Research Logistics Quarterly*, No. 1, 1954, pp. 301-312.

[46] Dantzig, G. B., "Recent Advances in Linear Programming", *Management Science*, Vol. 2, 1956, pp. 131-144.

[47] Beale, E. M. L., "On Minimizing a Convex Function Subject to Linear Inequalities", *Journal of the Royal Statistical Society* (B), Vol. 17, 1955, pp. 173-184.

[48] Frank, M. and P. Wolfe, "An Algorithm for Quadratic Programming", *Naval Research Logistics Quarterly*, Vol. 3, 1956, pp. 95-110.

[49] Wolfe, P., "The Simplex Method for Quadratic Programming", *Econometrica*, Vol. 27, No. 3, 1959, pp. 382-398.

[50] Arrow, K. J., L. Hurwicz, and H. Uzawa, *Studies in Linear and Non-Linear Programming*, Stanford University Press, Stanford, Calif., 1958.

[51] Lemke, C., "The Constrained Gradient Method of Linear Programming", *Journal of the Society for Industrial and Applied Mathematics*, Vol. 9, pp. 1-17.

[52] Rosen, J., "The Gradient Projection Method for Nonlinear Programming. Part I Linear Constraints", *Journal of the Society for Industrial and Applied Mathematics*, Vol. 9, 1960, pp. 181-217.

[53] Zoutendijk, G., *Methods of Feasible Directions*, Amsterdam: Elsevier, 1960.

[54] Davidon, W. C., "Variable Method for Minimization", Argonne Natl. Lab., ANL-5990 Rev., Univ. of Chicago, 1959.

[55] Doerfler, T. E., "PARTAN, Minimization by Method of Parallel Tangents", Iowa State University, Internal Memorandum, April (1964).

[56] Fletcher, R. and C. M. Reeves, "Function Minimization by Conjugate Gradients", *The Computer Journal*, Vol. 7, 1964, pp. 149-154.

[57] Powell, M. D. J., "An Efficient Method for Finding the Minimum of a Function of Several Variables without Calculating Derivatives", *The Computer Journal*, Vol. 7, 1964, pp. 155-162.

[58] Hooke, R. and T. A. Jeeves, "'Direct Search' Solution of Numerical and Statistical Problems", J. Assoc. for Comp. Mach., 1961, pp. 212-229.

[59] Karnopp, D. C., "Random Search Techniques for Optimization Problems", *Automatica*, Vol. 1, 1963, pp. 111-121.

[60] Brooks, S. H., "A Discussion of Random Methods for Seeking Maxima", *Operations Research*, Vol. 6, 1958, pp. 244-251.

[61] Wilde, D. J. and C. S. Beightler, *Foundations of Optimization*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1967.

[62] Hadley, G., *Linear Programming*, Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1962.

[63] __________, *Nonlinear and Dynamic Programming*, Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1964.

[64] Vajda, S., *Mathematical Programming*, Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1961

[65] Simmonard, M., *Linear Programming*, W. J. Jewell (translator), Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1966.

[66] Leon, A., "A Classical Bibliography on Optimization", in *Recent Advances in Optimization Techniques*, A. Lavi and t. P. Vogl (eds.), John Wiley & Sons, 1966.

[67] Abadie, J. (ed.), *Nonlinear Programming*, North-Holland Publishing Co., Amsterdam and John Wiley & Sons, Inc., New York, 1967.

[68] Karlin, S., *Mathematical Methods and Theory in Games, Programming, and Economics*, Vol. 1, Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1959.

[69] Balas, E., "An Infeasibility-Pricing Decomposition Method for Linear Programs", *Operations Research*, Vol. 14, 1966, pp. 847-873.

[70] Rosen, J. B., "Convex Partition Programming", in *Recent Advances in Mathematical Programming*, R. L. Graves and P. Wolfe (eds.), McGraw-Hill Book Company, Inc., New York, 1963.

[71] Balas, E., "An Additive Algorithm for Solving Linear Programs with Zero-One Variables", *Operations Research*, Vol. 13, 1965, pp. 517-546.

[72] ________, "Duality in Discrete Programming", *Technical Report No. 67-5*, Operations Research House, Stanford University, July 1967.

[73] Balinski, M., "Maximum Matching, Minimum Covering, and Duality", Sixth International Symposium on Mathematical Programming, Princeton, August, 1967

[74] Avriel, M. and D. J. Wilde, "Stochastic Geometric Programming", Sixth International Symposium on Mathematical Programming, Princeton, August 1967.

[75] Dreyfus, S. E., *Dynamic Programming and the Calculus of Variations*, The RAND Corporation, R-441-Pr, August, 1965.

BIOGRAPHICAL NOTE

Felipe Ochoa-Rosso was born on October 31, 1938, in Guadalajara, México. He received the degrees of Ingeniero Civil with "Mención Honorífica" in September 1962 and of Maestro en Ingeniería in August 1964, from the Universidad Nacional Autónoma de México. For his undergraduate work he received the "al Meritus" medal from the International Rotary Club, (México City), and upon his graduation the Colegio de Ingenieros Civiles de México bestowed upon him an award for being the "mejor pasante" of the School of Engineering class of 1956-1960.

On a Technical Fellowship from the French Government, the author spent the year 1962-1963 in Paris, studying computer systems with the Compagnie des Machines Bull. Upon his return from Paris he continued his work with Bull as their assistant director of the Scientific Department of the Mexico City branch.

The four years prior to commencing his doctoral program at the Massachusetts Institute of Technology in 1964, the author was a Professor of the Faculty of Engineering, National University of Mexico. Since 1966, as an Instructor of the Civil Engineering Department at M.I.T., he has been teaching courses on Optimization Theory and participating in development of a computer software system for an integrated facility for civil engineering. His doctoral program has focussed on three areas: systems engineering and operations research, computer science and transportation systems planning.

The author is a member of Sigma Xi, Chi Epsilon, the Operations Research Society of America, the Société des Ingeniéurs Civils de France, the Colegio de Ingenieros Civiles de México, and the Association for Computing Machinery

In August 1964, he married Sandra L. Rosellini of Seattle, Washington. They have two daughters, María Alejandra and Jacqucline Kate.

PUBLICATIONS

[1] *On the Theory of Thin Elastic Anisotropic Shells*, (in Spanish), C.E. Thesis, National University of Mexico, September 1962.

[2] "Programming Possibilities for Equilibrium Problems in Continuous Media", (in French), paper prepared for the Compagnie des Machines Bull, Paris, April 1963.

[3] *Computer Analysis of Earth Dam Stability*, (in Spanish), Master of C.E. Thesis, National University of Mexico, September 1964.

[4] "Mathematical Programming in the Optimum Design of Structures", (in Spanish), Professional Paper 13, Massachusetts Institute of Technology, Department of Civil Engineering, presented at the "Reunion Conjunta CICM-ASCE sobre Ingeniería Civil", Mexico City, February 1966.

[5] "The ICETRAN Language", in *ICES: Programmers' Reference Manual*, Jane C. Jordan (ed.), R67-50, Massachusetts Institute of Technology, Department of Civil Engineering, October 1967.

[6] *OPTECH User's Guide An Optimization Techniques Subsystem of ICES*, with David K. Bivins and Herve Thiriez, Massachusetts Institute of Technology, Department of Civil Engineering, October 1967.